Available online at www.sciencedirect.com
SCIENCE DIRECT®
Comput. Methods Appl. Mech. Engrg. 192 (2003) 3585–3618
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

A general framework for conservative single-step time-integration schemes with higher-order accuracy for a central-force system

E. Graham, G. Jelenić *,[1]

Department of Aeronautics, Imperial College of Science, Technology and Medicine, London SW7 2BY, UK

Received 30 December 2002; received in revised form 21 April 2003; accepted 8 May 2003

Dedicated to the memory of Mike Chrisfield – colleague, mentor and friend

Abstract

A general framework for algorithms that conserve angular momentum for single-body central-force problems is presented. It is shown that any family of momentum-conserving algorithms can have at most three free parameters, one of which may be used to ensure energy conservation (and hence will be configuration-dependent). Further restrictions can be made that enable the algorithms to recover the orbits of relative equilibria of the underlying physical problem. In addition, the algorithms can be made time-reversible, whilst still leaving two parameters unspecified. The order of accuracy of a general momentum-conserving family is analysed, and it is shown that energy–momentum algorithms that preserve the underlying physical relative equilibria can have unlimited accuracy if the two remaining parameters are appropriately chosen functions of the configuration and the time-step: this does not require any additional degrees of freedom, extra stages of calculation or information from past solutions. Numerical examples are given that show the performance of some representative higher-order schemes when applied to stiff and non-stiff problems, and the issue of Newton–Raphson convergence is discussed.

Keywords: Non-linear dynamics; Higher-order accuracy; Conservation properties; Relative equilibria; Time reversibility; Non-unique solutions; Implicit algorithms

1. Introduction

For modern-day numerical integration schemes to be considered successful when applied to a variety of non-linear Hamiltonian systems with symmetries, they need to conserve one or both of the first integrals of motion (namely the total energy and momenta) [1–9]. In [3,5,10] it was shown that for stiff dynamic

* Corresponding author.
E-mail address: g.jelenic@ic.ac.uk (G. Jelenić).
[1] Financially supported by EPSRC under contract AF/100089, which is gratefully acknowledged.

0045-7825/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0045-7825(03)00366-9

problems with large time-steps, algorithms that conserve the total energy perform better than those that preserve the symplectic structure of the Hamiltonian, and the two are known to be mutually exclusive for non-integrable problems [11]. An important consequence of the presence of symmetries in a Hamiltonian system is the existence of families of *relative equilibrium states*, and a strong case for algorithms to preserve these relative equilibria was made in [9,10].

It is also accepted that any viable time-integration algorithm should be at least second-order accurate, and it is usually agreed that such an algorithm should not need more than one set of implicit equations to be solved at each time-step [12]. For the central-force problem considered in this paper, a second-order accurate algorithm exists which conserves energy and angular momentum, preserves relative equilibrium states and is also time-reversible. It can be deduced from the algorithms of many authors [2–8] when applied to a single-body Hamiltonian system; for a description of this algorithm see, for example, Eqs. (6.1) and (6.4) of [10]. A generalisation of this algorithm that allows for high-frequency dissipation whilst keeping the total energy bounded was given in [9].

A natural question that arises is: Can we improve upon second-order accuracy whilst retaining all these other properties? The answer is currently yes, but usually at some extra cost. This issue was addressed by Tarnow and Simo [13], who proposed a general strategy for turning second-order algorithms into fourth-order algorithms without changing any other properties. This is achieved by computing intermediate results at two additional points in time for each time-step taken; thus the computational cost is three times higher for the new fourth-order scheme. A further disadvantage is that this procedure involves stepping backwards in time, using a larger time-step size than the original algorithm. This makes the principle less attractive for algorithms that are not time-reversible, and increases the risk of instability or divergence during the Newton–Raphson iteration. This strategy is actually a special case of those given independently by Yoshida [14] and Forest [15], whereby fourth-, sixth- and eighth-order algorithms can be developed from a second-order algorithm using 3, 8 and 16 intermediate results, respectively. A similar idea was used by Fung in [16] to produce higher-order algorithms for linear analysis that are based on Newmark's method; comparisons with the method of Tarnow and Simo are made therein. This idea was extended to non-linear analysis in [17], but the stability and accuracy properties of the resulting schemes were not proven in the non-linear regime.

An alternative approach to improving accuracy can be taken by discretising the equations of motion using finite elements in time, where the accuracy can be prescribed by the degree of the polynomial basis functions chosen. Fourth-order momentum-conserving algorithms using continuous time finite elements were developed by Betsch and Steinmann [18]. Similarly, third-order schemes with dissipative characteristics derived from a discontinuous finite element formulation were presented by Fan et al. [19] for linear dynamics and Bauchau and Joo [20] for non-linear dynamics. Each of these schemes involves additional degrees of freedom at each time-step; at the mid-point of the time interval for the conserving schemes, and to cater for the discontinuities at the end-points for the dissipative schemes. For a Hamiltonian system, the number of equations to be solved is twice that of a standard time-integration algorithm.

Higher-order accurate algorithms can also be based on Taylor series expansions of the state variables. Historically, Adams methods have been used that approximate the derivatives with finite difference formulae (see e.g. [21]). However, these become multi-step methods when going beyond second-order accuracy, and suffer the usual drawbacks of needing a special starting procedure and having to store the solutions at previous time-steps. For the equations of motion, these derivatives have a simple form, and can thus be expressed directly without need of approximation. Early work along these lines was done by Argyris et al. [22,23], who presented *arbitrarily* accurate algorithms that are time-reversible, although not conservative. They were followed by LaBudde and Greenspan [1] who produced arbitrarily accurate schemes that also conserve energy and angular momentum for a central-force problem. These schemes are not time-reversible, however, and do not preserve the orbits of relative equilibria when higher than second-order accurate.

We note the absence of a conservative, time-reversible algorithm that preserves the orbits of relative equilibria, with an order of accuracy greater than two, that does not involve additional computational expense of some description. The aim of this paper, therefore, is to provide a framework in which higher-order algorithms with all of these properties can be developed at no extra cost. The theory is set out in Sections 2–5, and examples of existing algorithms that fit into the framework are given in Section 6. Numerical results showing the performance of some representative higher-order schemes when applied to a model problem are shown in Sections 7 and 8.

2. Equations of motion

The equations of motion for a single-body central-force dynamical system can be described using the Hamiltonian formulation as

$$\begin{aligned} \dot{\boldsymbol{q}} &= \nabla_{\boldsymbol{p}} H(\boldsymbol{q},\boldsymbol{p}), \\ \dot{\boldsymbol{p}} &= -\nabla_{\boldsymbol{q}} H(\boldsymbol{q},\boldsymbol{p}), \end{aligned} \tag{2.1}$$

where $\boldsymbol{q},\boldsymbol{p} \in \mathbb{R}^3$ denote the position and momentum, respectively, of the body with respect to an origin O of an inertial frame, and $H(\cdot,\cdot)$ is the Hamiltonian function representing the total energy of the system, with a superimposed dot indicating a time derivative. $H(\boldsymbol{q},\boldsymbol{p})$ is defined as

$$H(\boldsymbol{q},\boldsymbol{p}) = V(\boldsymbol{q}) + T(\boldsymbol{p}), \tag{2.2}$$

where $V(\cdot)$ and $T(\cdot)$ represent the potential and kinetic energies of the system, respectively. We shall presume the existence of a unique solution to (2.1).

For a body of mass m moving in a central force field of origin O, we have

$$V(\boldsymbol{q}) = \widetilde{V}(l) \quad \text{and} \quad T(\boldsymbol{p}) = \frac{1}{2m}\boldsymbol{p}\cdot\boldsymbol{p}, \tag{2.3}$$

where $l = \|\boldsymbol{q}\|$ and $\|\cdot\|$ denotes the Euclidean (or two-) norm; thus the potential function $V(\boldsymbol{q})$ is dependent only on the magnitude of the vector $\boldsymbol{q}$. So we have

$$\nabla_{\boldsymbol{q}} H(\boldsymbol{q},\boldsymbol{p}) = \nabla V(\boldsymbol{q}) = \widetilde{V}'(l)\nabla\{\sqrt{\boldsymbol{q}\cdot\boldsymbol{q}}\} = \frac{\widetilde{V}'(l)}{l}\boldsymbol{q}$$

and

$$\nabla_{\boldsymbol{p}} H(\boldsymbol{q},\boldsymbol{p}) = \frac{1}{m}\boldsymbol{p}$$

and hence our equations of motion become

$$\begin{aligned} \dot{\boldsymbol{q}} &= \frac{1}{m}\boldsymbol{p}, \\ \dot{\boldsymbol{p}} &= -\frac{\widetilde{V}'(l)}{l}\boldsymbol{q}. \end{aligned} \tag{2.4}$$

Standard properties of a Hamiltonian system defined by (2.1) are conservation of the Hamiltonian H and preservation of the symplectic two-form (see e.g. [4] for details). An additional property of the *central-force* Hamiltonian problem defined by (2.2) and (2.3) is conservation of the total angular momentum $\mathscr{J} = \boldsymbol{q}\times\boldsymbol{p}$, which in turn implies that a set of relative equilibrium states exist as possible solutions. These states are characterised by circular orbits of radius l_0 along which the mass moves with a constant angular velocity $w_0 = \sqrt{\widetilde{V}'(l_0)/(ml_0)}$ (see e.g. [9,10] for further explanation).

3. Algorithm derivation

We wish to generate a family of single-step algorithms to solve system (2.4) approximately, that can then be specialised to conserve various constants of motion. The general form for such a family is

$$\begin{aligned} \boldsymbol{q}_{n+1} &= a\boldsymbol{q}_n + b\boldsymbol{p}_n, \\ \boldsymbol{p}_{n+1} &= c\boldsymbol{q}_n + d\boldsymbol{p}_n, \end{aligned} \tag{3.1}$$

where $\boldsymbol{q}_k, \boldsymbol{p}_k \in \mathbb{R}^3$ are the discrete approximations to the positions $\boldsymbol{q}(t_k)$ and momenta $\boldsymbol{p}(t_k)$ at time $t_k \geqslant 0$. The quantities a, b, c and $d \in \mathbb{R}$ are unspecified parameters that may depend upon $\boldsymbol{p}_n$, $\boldsymbol{q}_n$, $\boldsymbol{p}_{n+1}$, $\boldsymbol{q}_{n+1}$ and Δt where $\Delta t = t_{n+1} - t_n$ is the (non-zero) time-step length. By defining

$$\boldsymbol{z}_n = \left\{ \begin{matrix} \boldsymbol{q}_n \\ \boldsymbol{p}_n \end{matrix} \right\}$$

we can naturally express algorithm (3.1) in matrix form as

$$\boldsymbol{z}_{n+1} = \boldsymbol{B}_{n+1}\boldsymbol{z}_n \quad \text{where } \boldsymbol{B}_{n+1} \equiv \boldsymbol{B}(\boldsymbol{z}_{n+1}, \boldsymbol{z}_n, \Delta t) = \begin{pmatrix} a\boldsymbol{I}_3 & b\boldsymbol{I}_3 \\ c\boldsymbol{I}_3 & d\boldsymbol{I}_3 \end{pmatrix} \tag{3.2}$$

with $\boldsymbol{I}_3 \in \mathbb{R}^{3\times 3}$ the identity matrix. Note that, to prevent the possible occurrence of the solution $\boldsymbol{z}_{n+1} = \boldsymbol{0}$ without $\boldsymbol{z}_n = \boldsymbol{0}$, we require $\boldsymbol{B}_{n+1}$ to be non-singular. From $(3.2)_2$ we have

$$\det(\boldsymbol{B}_{n+1}) = (ad - bc)^3, \tag{3.3}$$

thus we may proceed with arbitrary parameters a, b, c and d subject to the condition

$$ad - bc \neq 0. \tag{3.4}$$

We emphasise that algorithm (3.1) is a non-linear algorithm in general, even though Eq. $(3.2)_1$ appears to be linear: the non-linearity is expressed through those parameters among a, b, c and d which depend (non-linearly) on $\boldsymbol{p}_{n+1}$ and $\boldsymbol{q}_{n+1}$.

3.1. Inherent angular momentum conservation

We now consider only the instances of algorithm (3.1) that provide conservation of the angular momentum $\boldsymbol{\mathcal{J}} = \boldsymbol{q} \times \boldsymbol{p}$ i.e. $\boldsymbol{\mathcal{J}}_{n+1} = \boldsymbol{\mathcal{J}}_n$. From (3.1) we have

$$\boldsymbol{\mathcal{J}}_{n+1} - \boldsymbol{\mathcal{J}}_n = \boldsymbol{q}_{n+1} \times \boldsymbol{p}_{n+1} - \boldsymbol{q}_n \times \boldsymbol{p}_n = (a\boldsymbol{q}_n + b\boldsymbol{p}_n) \times (c\boldsymbol{q}_n + d\boldsymbol{p}_n) - \boldsymbol{q}_n \times \boldsymbol{p}_n = (ad - bc - 1)\boldsymbol{q}_n \times \boldsymbol{p}_n,$$

therefore algorithm (3.1) will conserve angular momentum in general if and only if

$$ad - bc = 1 \quad \text{i.e. } \det(\boldsymbol{B}_{n+1}) = 1. \tag{3.5}$$

We note that this condition automatically assures that $\boldsymbol{B}_{n+1}$ is non-singular, as shown by (3.3) and (3.4). Eq. (3.5) therefore fixes one of the parameters a, b, c and d: thus *any family of single-step algorithms that conserves angular momentum can have at most three free parameters.*

3.2. Choice of parameters

Given the multitude of ways to express the three-parameter momentum-conserving family of algorithms given by (3.1) and (3.5), we choose one that will relate easily to previous work on this subject. We first define, for any given quantity $(\cdot)$, the notation

$$(\cdot)_\Delta := (\cdot)_{n+1} - (\cdot)_n \quad \text{and} \quad (\cdot)_\alpha := [1-\alpha](\cdot)_n + \alpha(\cdot)_{n+1} \quad \text{for } \alpha \in \mathbb{R}.$$

From [24], we have the momentum-conserving family

$$\begin{aligned}\frac{\beta}{\Delta t}\boldsymbol{q}_\Delta &= \frac{1}{m}\boldsymbol{p}_{1-\alpha},\\ \frac{\beta}{\Delta t}\boldsymbol{p}_\Delta &= -\xi\boldsymbol{q}_\alpha,\end{aligned} \tag{3.6}$$

where α, β and ξ are the free parameters expressible in terms of a, b, c and d with $ad - bc = 1$. This approximates the continuous system (2.4), with $\frac{1}{\Delta t}\boldsymbol{q}_\Delta$ and $\frac{1}{\Delta t}\boldsymbol{p}_\Delta$ representing $\dot{\boldsymbol{q}}$ and $\dot{\boldsymbol{p}}$, respectively.

In this work, we choose to express our family of momentum-conserving algorithms as

$$\begin{aligned}\frac{1}{\Delta t}(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}) &= \frac{1}{m}\boldsymbol{p}_{1/2},\\ \frac{1}{\Delta t}(\beta\boldsymbol{p}_\Delta + \gamma\boldsymbol{p}_{1/2}) &= -\xi\boldsymbol{q}_{1/2},\end{aligned} \tag{3.7}$$

where β, γ and ξ are now the free parameters. We will refer to this family of algorithms collectively as Algorithm 1, and it is equivalent to (3.1) with the parameter relations

$$a = \frac{\left(\beta + \frac{1}{2}\gamma\right)^2 - \frac{1}{4m}\xi\Delta t^2}{\mathscr{D}}, \quad b = \frac{\frac{1}{m}\beta\Delta t}{\mathscr{D}}, \quad c = -\frac{\xi\beta\Delta t}{\mathscr{D}} \quad \text{and} \quad d = \frac{\left(\beta - \frac{1}{2}\gamma\right)^2 - \frac{1}{4m}\xi\Delta t^2}{\mathscr{D}}$$
$$\text{where } \mathscr{D} = \beta^2 - \frac{1}{4}\gamma^2 + \frac{1}{4m}\xi\Delta t^2 \neq 0. \tag{3.8}$$

The chosen parameters β, γ and ξ will now be determined by conservation criteria and local accuracy considerations. Given that (3.8) involves quotients in β, γ and ξ, it will sometimes be more convenient to write (3.2) in the form

$$\mathscr{D}\boldsymbol{z}_{n+1} = \widehat{\boldsymbol{B}}_{n+1}\boldsymbol{z}_n \quad \text{where } \widehat{\boldsymbol{B}}_{n+1} = \mathscr{D}\boldsymbol{B}_{n+1} := \begin{pmatrix} \widehat{B}_{11}\boldsymbol{I}_3 & \widehat{B}_{12}\boldsymbol{I}_3 \\ \widehat{B}_{21}\boldsymbol{I}_3 & \widehat{B}_{22}\boldsymbol{I}_3 \end{pmatrix} \tag{3.9}$$

and $\widehat{B}_{11} = a\mathscr{D}$, $\widehat{B}_{12} = b\mathscr{D}$, $\widehat{B}_{21} = c\mathscr{D}$ and $\widehat{B}_{22} = d\mathscr{D}$.

3.3. Existence and uniqueness of solutions

We now give the condition under which a solution $\boldsymbol{z}_{n+1}$ to Eq. $(3.2)_1$ may be found. Criteria for the existence and uniqueness of solutions to general non-linear equations can be found in [25, pp. 36–38], and a discussion relating these to scalar difference equations is given in [26, pp. 215–217]. From these, we derive the *Lipschitz condition*

$$\|[\boldsymbol{B}(\tilde{\boldsymbol{x}}, \boldsymbol{z}_n, \Delta t) - \boldsymbol{B}(\hat{\boldsymbol{x}}, \boldsymbol{z}_n, \Delta t)]\boldsymbol{z}_n\| < \|\tilde{\boldsymbol{x}} - \hat{\boldsymbol{x}}\| \quad \forall \tilde{\boldsymbol{x}}, \hat{\boldsymbol{x}} \in \mathbb{R}^6 \tag{3.10}$$

which, if satisfied, implies the existence of a unique $\boldsymbol{z}_{n+1}$ satisfying $(3.2)_1$. Fulfilment of (3.10) depends on a, b, c, d and Δt, since these define $\boldsymbol{B}$, and also on $\boldsymbol{z}_n$. Provided that Algorithm 1 is *convergent*, i.e. it recovers the solution to system (2.4) as $\Delta t \to 0$, we can guarantee that (3.10) will be satisfied for sufficiently small yet non-zero Δt (since (2.4) is presumed to have a unique solution). Hence there exists $\Delta t_{\text{cr}} \in \mathbb{R}^+$ such that a convergent algorithm will yield a unique solution for $\boldsymbol{p}_{n+1}$ and $\boldsymbol{q}_{n+1}$ for all $\Delta t \leqslant \Delta t_{\text{cr}}$.

In general, we will not know the critical value of Δt necessary to provide unique $\boldsymbol{p}_{n+1}, \boldsymbol{q}_{n+1} \in \mathbb{R}^3$ for all $n \in \mathbb{Z}^+$ given $\boldsymbol{p}_0$ and $\boldsymbol{q}_0$. Practically speaking, we may expect that a given Δt selected for accuracy requirements will be sufficient to ensure a unique solution at all time-steps, but we emphasise that this is not guaranteed.

4. Properties of the algorithm

We will now derive conditions for β, γ and ξ under which Algorithm 1 will conserve energy, preserve the orbits of relative equilibria of the underlying physical system and be time-reversible.

4.1. Conservation of energy

The discrete total energy at time-step n is given by the Hamiltonian function $H_n := H(\boldsymbol{q}_n, \boldsymbol{p}_n)$. From (2.2) and (2.3) we can write

$$H_n = \widetilde{V}_n + \frac{1}{2m}\boldsymbol{p}_n \cdot \boldsymbol{p}_n, \tag{4.1}$$

where $V_n := V(\boldsymbol{q}_n) = \widetilde{V}(l_n)$ is the discrete potential energy.

Proposition 1. *Algorithm* 1 *conserves energy if*

$$\xi = \frac{\beta\widetilde{V}_\Delta - \dfrac{m}{\Delta t^2}\gamma\|\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}\|^2}{(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2})\cdot\boldsymbol{q}_{1/2}} \quad \textit{where } \beta \neq 0 \tag{4.2}$$

provided that the right-hand side of (4.2) *is always well defined.*

The proof is given in Appendix A.1. Eq. (4.2) therefore fixes one of the parameters β, γ and ξ, for $\beta \neq 0$. As done in [24], we assign ξ to ensure energy conservation, so that (4.2) is solved as an equation in ξ. This implies that *any family of single-step energy–momentum algorithms can have at most two free parameters*, since ξ has now been put to use. We note that for $\gamma = 0$, (4.2) becomes

$$\xi = \frac{\widetilde{V}_\Delta}{\boldsymbol{q}_\Delta \cdot \boldsymbol{q}_{1/2}} = \frac{\widetilde{V}_\Delta}{\frac{1}{2}(l_{n+1}^2 - l_n^2)}. \tag{4.3}$$

It can be seen that for ξ defined by (4.2) we have $\beta = 0 \Rightarrow \mathscr{D} = 0$, for $\mathscr{D}$ as defined in (3.8). Thus the requirement $\beta \neq 0$ is encapsulated in the condition $\mathscr{D} \neq 0$ for energy-conserving schemes. We shall now assume $\beta \neq 0$ holds unless otherwise stated.

Remark 1. An equivalent energy conservation condition can be derived for the family defined by (3.6) which becomes a quadratic equation in ξ with coefficients in terms of α and β. However, a real solution for ξ is not guaranteed for arbitrary α [24]. An alternative equation to (4.2) can also be derived for Algorithm 1 by expressing $H_{n+1} - H_n$ in terms of $\boldsymbol{q}_n$ and $\boldsymbol{p}_n$ instead of $\boldsymbol{q}_{n+1}$ and $\boldsymbol{q}_n$; in other words, by writing $\boldsymbol{p}_{n+1}$ in terms of $\boldsymbol{q}_n$ and $\boldsymbol{p}_n$. This equation is still implicitly dependent on $\boldsymbol{q}_{n+1}$ through $\widetilde{V}_{n+1}$, and is also quadratic in ξ [27]. In [1], LaBudde and Greenspan arrive at an equation for energy conservation using this second approach; see Eq. (4.46) of [1] and also Section 6 here.

4.2. Preservation of relative equilibria

A relative equilibrium state (or *steady-state*) occurs when the initial conditions $\{\boldsymbol{q}_0, \boldsymbol{p}_0\}$ for Algorithm 1 are such that

$$\boldsymbol{q}_0 \cdot \boldsymbol{p}_0 = 0 \quad \text{and} \quad \frac{1}{m}\|\boldsymbol{p}_0\|^2 = \widetilde{V}'(\|\boldsymbol{q}_0\|)\|\boldsymbol{q}_0\|. \tag{4.4}$$

Since we have the relationship $mw\|\boldsymbol{q}\|^2 = \|\boldsymbol{q} \times \boldsymbol{p}\|$ for the angular velocity w, we can express $(4.4)_2$ as

$$mw_0^2 = \frac{\widetilde{V}'(\|\boldsymbol{q}_0\|)}{\|\boldsymbol{q}_0\|}, \tag{4.5}$$

where w_0 is the initial angular velocity. To preserve orbits of relative equilibria, therefore, an algorithm must ensure that whenever (4.4) holds, we have

$$\|\boldsymbol{q}_n\| = \|\boldsymbol{q}_0\|, \quad \|\boldsymbol{p}_n\| = \|\boldsymbol{p}_0\| \quad \text{and} \quad \boldsymbol{q}_n \cdot \boldsymbol{p}_n = 0 \tag{4.6}$$

for all n, giving $\frac{1}{m}\|\boldsymbol{p}_n\|^2 = \widetilde{V}'(\|\boldsymbol{q}_n\|)\|\boldsymbol{q}_n\|$. Introducing the abbreviation

$$f_n := \frac{\widetilde{V}'(\|\boldsymbol{q}_n\|)}{\|\boldsymbol{q}_n\|}$$

we see that by induction on n, (4.4) and (4.6) are equivalent to the condition that

$$\boldsymbol{q}_n \cdot \boldsymbol{p}_n = 0 \text{ and } \frac{1}{m}\|\boldsymbol{p}_n\|^2 = f_n\|\boldsymbol{q}_n\|^2 \;\Rightarrow\; \|\boldsymbol{q}_{n+1}\| = \|\boldsymbol{q}_n\|,\ \|\boldsymbol{p}_{n+1}\| = \|\boldsymbol{p}_n\| \text{ and } \boldsymbol{q}_{n+1} \cdot \boldsymbol{p}_{n+1} = 0. \tag{4.7}$$

In other words, relative equilibrium conditions at time-step n should imply the same conditions at time-step $n+1$. We now introduce, for any given quantity $(\cdot)$, the notation

$$(\cdot)^{\mathrm{RE}} := \lim_{\substack{\|\boldsymbol{q}_{n+1}\| \to \|\boldsymbol{q}_n\|, \\ \|\boldsymbol{p}_{n+1}\| \to \|\boldsymbol{p}_n\|, \\ \boldsymbol{q}_{n+1} \cdot \boldsymbol{p}_{n+1} \to 0}} \left\{ (\cdot) \Big|_{\boldsymbol{q}_n \cdot \boldsymbol{p}_n = 0,\ \frac{1}{m}\|\boldsymbol{p}_n\|^2 = f_n\|\boldsymbol{q}_n\|^2} \right\}.$$

Proposition 2. *Algorithm* 1 *preserves the orbits of relative equilibria provided that*

$$\beta^{\mathrm{RE}}\gamma^{\mathrm{RE}} = 0 \quad \textit{and} \quad \beta^{\mathrm{RE}}(f_n - \xi^{\mathrm{RE}}) = 0 \tag{4.8}$$

and that it gives a unique solution for $\boldsymbol{p}_{n+1}$ *and* $\boldsymbol{q}_{n+1}$.

The proof is given in Appendix A.2. Note that the requirement that $\boldsymbol{p}_{n+1}$ and $\boldsymbol{q}_{n+1}$ be unique suggests that preservation of relative equilibria for any algorithm may be dependent on the size of the time-step used (namely one that ensures the solutions are unique).

Remark 2. The fact that $\boldsymbol{p}_{n+1}$ and $\boldsymbol{q}_{n+1}$ must be uniquely determined to assure that an algorithm preserves relative equilibria has been implicitly used in similar proofs by other authors (e.g. [9]). Note that by contrast, our conditions for energy conservation did not require the solutions to be unique: should there be multiple choices for $\boldsymbol{p}_{n+1}$ and $\boldsymbol{q}_{n+1}$, they would all keep energy conserved provided the conditions were satisfied.

4.3. Conservation of energy and the preservation of relative equilibria

Lemma 1. *Any algorithm that preserves the orbits of relative equilibria also conserves energy along the orbits of relative equilibria.*

Proof. This is self-evident from (4.1) and (4.7). □

Lemma 1 assures that no conflict arises from having conservation of energy and preservation of relative equilibria within the same algorithm. It does not guarantee that an energy-conserving algorithm will

preserve relative equilibria, however, and indeed generally speaking it will not. For this to be guaranteed, Eqs. (4.2) and (4.8) must be satisfied simultaneously, i.e.

$$\xi^{\mathrm{RE}} \equiv \left[\frac{\beta \widetilde{V}_\Delta - \frac{m}{\Delta t^2}\gamma \|\beta \boldsymbol{q}_\Delta - \gamma \boldsymbol{q}_{1/2}\|^2}{(\beta \boldsymbol{q}_\Delta - \gamma \boldsymbol{q}_{1/2}) \cdot \boldsymbol{q}_{1/2}} \right]^{\mathrm{RE}} = f_n \quad \text{and} \quad \gamma^{\mathrm{RE}} = 0. \tag{4.9}$$

Note that there are many possible definitions for γ that fulfil $(4.9)_2$: for example $\gamma := 0$, $\gamma := \boldsymbol{p}_n \cdot \boldsymbol{q}_n + \boldsymbol{p}_{n+1} \cdot \boldsymbol{q}_{n+1}$ or $\gamma := \tilde{\gamma}(l_{n+1} - l_n)$ for non-zero $\tilde{\gamma}$ all give $\gamma^{\mathrm{RE}} = 0$. It is instructive to analyse what happens to ξ^{RE} in each of these cases.

In the case that γ is identically zero, ξ is defined by (4.3), and thus $(4.9)_1$ stipulates that

$$\left[\frac{\widetilde{V}_\Delta}{\frac{1}{2}(l_{n+1}^2 - l_n^2)} \right]^{\mathrm{RE}} = f_n. \tag{4.10}$$

Since

$$\left[\frac{\widetilde{V}_\Delta}{\frac{1}{2}(l_{n+1}^2 - l_n^2)} \right]^{\mathrm{RE}} \equiv \lim_{l_{n+1} \to l_n} \left\{ \frac{\widetilde{V}(l_{n+1}) - \widetilde{V}(l_n)}{\frac{1}{2}(l_{n+1} - l_n)(l_{n+1} + l_n)} \right\} = \frac{\widetilde{V}'(l_n)}{l_n}$$

we see that $(4.9)_1$ is *always* satisfied if $\gamma := 0$.

In the second case $\gamma := \boldsymbol{p}_n \cdot \boldsymbol{q}_n + \boldsymbol{p}_{n+1} \cdot \boldsymbol{q}_{n+1}$, the expression for ξ^{RE} in $(4.9)_1$ is indeterminate, since the numerator and denominator both tend to zero with no obvious limiting value for the quotient. Therefore this is *not* a suitable definition for γ, since the necessary condition on ξ cannot then be established.

The case $\gamma := \tilde{\gamma}(l_{n+1} - l_n)$ is very interesting. Here we have, from (4.2),

$$\xi = \frac{\beta \widetilde{V}_\Delta - \frac{m}{\Delta t^2}\tilde{\gamma}(l_{n+1} - l_n)\|\beta \boldsymbol{q}_\Delta - \gamma \boldsymbol{q}_{1/2}\|^2}{[\beta \boldsymbol{q}_\Delta - \tilde{\gamma}(l_{n+1} - l_n)\boldsymbol{q}_{1/2}] \cdot \boldsymbol{q}_{1/2}}.$$

The numerator can be expressed as $(l_{n+1} - l_n)(\beta \widetilde{V}_D - \frac{m}{\Delta t^2}\tilde{\gamma}\|\beta \boldsymbol{q}_\Delta - \gamma \boldsymbol{q}_{1/2}\|^2)$ where

$$\widetilde{V}_D := \frac{\widetilde{V}_\Delta}{l_{n+1} - l_n}$$

and the denominator can be written as $(l_{n+1} - l_n)[\frac{1}{2}\beta(l_{n+1} + l_n) - \tilde{\gamma}\|\boldsymbol{q}_{1/2}\|^2]$. Thus we can cancel the factor $(l_{n+1} - l_n)$, leaving

$$\xi = \frac{\beta \widetilde{V}_D - \frac{m}{\Delta t^2}\tilde{\gamma}\|\beta \boldsymbol{q}_\Delta - \gamma \boldsymbol{q}_{1/2}\|^2}{\frac{1}{2}\beta(l_{n+1} + l_n) - \tilde{\gamma}\|\boldsymbol{q}_{1/2}\|^2}.$$

Since $\gamma^{\mathrm{RE}} = 0$ and $[\widetilde{V}_D]^{\mathrm{RE}} = \widetilde{V}'(l_n)$, this gives us

$$\xi^{\mathrm{RE}} \equiv \left[\frac{\beta \widetilde{V}_D - \frac{m}{\Delta t^2}\tilde{\gamma}\|\beta \boldsymbol{q}_\Delta - \gamma \boldsymbol{q}_{1/2}\|^2}{\frac{1}{2}\beta(l_{n+1} + l_n) - \tilde{\gamma}\|\boldsymbol{q}_{1/2}\|^2} \right]^{\mathrm{RE}} = \frac{\beta^{\mathrm{RE}} \widetilde{V}'(l_n) - \frac{m}{\Delta t^2}\tilde{\gamma}^{\mathrm{RE}}\|\beta^{\mathrm{RE}} \boldsymbol{q}_\Delta\|^2}{\beta^{\mathrm{RE}} l_n - \tilde{\gamma}^{\mathrm{RE}}\|\boldsymbol{q}_{1/2}\|^2} \tag{4.11}$$

provided $\beta^{\mathrm{RE}} l_n - \tilde{\gamma}^{\mathrm{RE}}\|\boldsymbol{q}_{1/2}\|^2 \neq 0$. If $\tilde{\gamma}^{\mathrm{RE}} = 0$, then (4.11) is identical to $(4.9)_1$ for $\beta^{\mathrm{RE}} \neq 0$: thus $(4.9)_1$ is always satisfied once again. On the other hand, if $\tilde{\gamma}^{\mathrm{RE}} \neq 0$, substituting (4.11) into $(4.9)_1$ and rearranging gives us

$$\beta^{\mathrm{RE}} \widetilde{V}'(l_n) - \frac{m}{\Delta t^2}\tilde{\gamma}^{\mathrm{RE}}(\beta^{\mathrm{RE}})^2\|\boldsymbol{q}_\Delta\|^2 = \beta^{\mathrm{RE}} \widetilde{V}'(l_n) - \tilde{\gamma}^{\mathrm{RE}}\|\boldsymbol{q}_{1/2}\|^2 \frac{\widetilde{V}'(l_n)}{l_n}$$

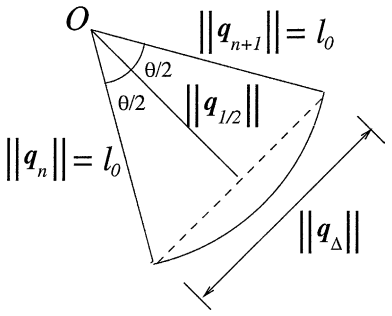

Fig. 1. Sector of area traced out during relative equilibrium motion.

which then leads to

$$\beta^{\mathrm{RE}} = \sqrt{\frac{\widetilde{V}'(l_n)}{m l_n}} \frac{\|\boldsymbol{q}_{1/2}\| \Delta t}{\|\boldsymbol{q}_\Delta\|}. \tag{4.12}$$

From (4.5) we have $w_0^2 = \widetilde{V}'(l_n)/(m l_n)$ (since $\|\boldsymbol{q}_n\| = \|\boldsymbol{q}_0\| \ \forall n$), and we see from Fig. 1 that $\tan(\frac{1}{2}\theta) = \frac{1}{2}\|\boldsymbol{q}_\Delta\|/\|\boldsymbol{q}_{1/2}\|$ where θ is the incremental angle defined by $\boldsymbol{q}_n \cdot \boldsymbol{q}_{n+1} = l_n l_{n+1} \cos\theta$. Thus (4.12) can be written as

$$\beta^{\mathrm{RE}} = \frac{\frac{1}{2} w_0 \Delta t}{\tan\left(\frac{1}{2}\theta\right)} \tag{4.13}$$

which is identical to Eq. (3.12) of [24]. This equation gives the criterion for the recovery of the *exact solution* to the steady-state problem defined by the initial conditions given in (4.4), as discussed in Section 3 of [24]. Hence for $\gamma := \tilde{\gamma}(l_{n+1} - l_n)$ where $\tilde{\gamma}^{\mathrm{RE}}$ is non-zero, (4.9) is satisfied if and only if (4.13) holds, which means that *if the orbits of relative equilibria are to be preserved, the exact solution for the steady-state problem will be recovered.*

4.4. Time reversibility

An algorithm is described as time-reversible if, at any given configuration $\boldsymbol{z}_{n+1}$, applying a negative time-step of $-\Delta t$ recovers the previous configuration $\boldsymbol{z}_n$ [14]. From (3.2), an algorithm is time-reversible if

$$\boldsymbol{z}_{n+1} = \boldsymbol{B}(\boldsymbol{z}_{n+1}, \boldsymbol{z}_n, \Delta t)\boldsymbol{z}_n \iff \boldsymbol{z}_n = \boldsymbol{B}(\boldsymbol{z}_n, \boldsymbol{z}_{n+1}, -\Delta t)\boldsymbol{z}_{n+1}. \tag{4.14}$$

We now introduce for any quantity $(\cdot)$ the notation

$$(\cdot)^{\mathrm{TR}} := (\cdot)|_{\boldsymbol{z}_{n+1} \leftrightarrow \boldsymbol{z}_n, \Delta t \leftrightarrow -\Delta t}.$$

Therefore from (3.2) we see that

$$\boldsymbol{B}_{n+1}^{\mathrm{TR}} := \boldsymbol{B}(\boldsymbol{z}_n, \boldsymbol{z}_{n+1}, -\Delta t) = \begin{pmatrix} a^{\mathrm{TR}}\boldsymbol{I}_3 & b^{\mathrm{TR}}\boldsymbol{I}_3 \\ c^{\mathrm{TR}}\boldsymbol{I}_3 & d^{\mathrm{TR}}\boldsymbol{I}_3 \end{pmatrix}. \tag{4.15}$$

Proposition 3. *Algorithm* 1 *is time-reversible if*

$$\beta = \beta^{\mathrm{TR}}, \quad \gamma = -\gamma^{\mathrm{TR}} \quad \textit{and} \quad \xi = \xi^{\mathrm{TR}}. \tag{4.16}$$

The proof is given in Appendix A.3.

5. Local accuracy analysis

We now analyse the *local* accuracy characteristics of Algorithm 1, and investigate its capacity for higher-order accuracy when applied to general non-linear problems.

5.1. Order of accuracy

We define the *local error vector* as

$$\boldsymbol{\epsilon} := \boldsymbol{z}_{n+1} - \boldsymbol{z}(t_{n+1}) \quad \text{when } \boldsymbol{z}_n = \boldsymbol{z}(t_n), \tag{5.1}$$

where $\boldsymbol{z}^{\mathrm{T}} = [\,\boldsymbol{q}^{\mathrm{T}} \quad \boldsymbol{p}^{\mathrm{T}}\,]$ as before. In other words, the local error represents the departure from the exact solution at a given time after one time-step; the particular point t_n is not significant. Throughout this section, we will assume the solution at time-step n to be exact, i.e. $\boldsymbol{z}_n = \boldsymbol{z}(t_n)$.

For local accuracy analysis, we must determine the dependence of $\boldsymbol{\epsilon}$ on Δt, and see how quickly $\|\boldsymbol{\epsilon}\| \to 0$ as $\Delta t \to 0$ and under what conditions. This is most simply accomplished by first defining the *residual vector*

$$\boldsymbol{g}(\boldsymbol{x}) := \widehat{\boldsymbol{B}}(\boldsymbol{x}, \boldsymbol{z}_n, \Delta t)\boldsymbol{z}_n - \mathscr{D}(\boldsymbol{x}, \boldsymbol{z}_n, \Delta t)\boldsymbol{x}, \tag{5.2}$$

where $\mathscr{D}$ and $\widehat{\boldsymbol{B}}$ were defined in (3.8) and (3.9), with

$$\mathscr{D}(\boldsymbol{z}_{n+1}, \boldsymbol{z}_n, \Delta t)\boldsymbol{z}_{n+1} = \widehat{\boldsymbol{B}}(\boldsymbol{z}_{n+1}, \boldsymbol{z}_n, \Delta t)\boldsymbol{z}_n.$$

From $(5.1)_1$ we can write

$$\boldsymbol{g}[\boldsymbol{z}(t_{n+1})] = \boldsymbol{g}(\boldsymbol{z}_{n+1} - \boldsymbol{\epsilon}) = \boldsymbol{g}(\boldsymbol{z}_{n+1}) - \nabla\boldsymbol{g}(\boldsymbol{z}_{n+1})\boldsymbol{\epsilon} + \mathscr{O}(\|\boldsymbol{\epsilon}\|^2)$$

using a Taylor expansion about $\boldsymbol{z}_{n+1}$, where $\nabla\boldsymbol{g}$ is the *Jacobian matrix* with components $\dfrac{\partial g_i}{\partial x_j}$. From (3.9) we have $\boldsymbol{g}(\boldsymbol{z}_{n+1}) = \boldsymbol{0}$, leaving

$$\boldsymbol{g}[\boldsymbol{z}(t_{n+1})] = -\nabla\boldsymbol{g}(\boldsymbol{z}_{n+1})\boldsymbol{\epsilon} + \mathscr{O}(\|\boldsymbol{\epsilon}\|^2) \tag{5.3}$$

which relates $\boldsymbol{g}[\boldsymbol{z}(t_{n+1})]$ to $\boldsymbol{\epsilon}$; thus analysis of $\boldsymbol{\epsilon}$ can be done via $\boldsymbol{g}[\boldsymbol{z}(t_{n+1})]$.

Analysing $\boldsymbol{g}[\boldsymbol{z}(t_{n+1})]$ is straightforward (if a little involved); from (5.2) we have

$$\boldsymbol{g}[\boldsymbol{z}(t_{n+1})] = \widehat{\boldsymbol{B}}[\boldsymbol{z}(t_{n+1}), \boldsymbol{z}_n, \Delta t]\boldsymbol{z}_n - \mathscr{D}[\boldsymbol{z}(t_{n+1}), \boldsymbol{z}_n, \Delta t]\boldsymbol{z}(t_{n+1}). \tag{5.4}$$

Introducing the abbreviation

$$\boldsymbol{f}(\boldsymbol{\zeta}) := \boldsymbol{f}[\boldsymbol{z}(t_{n+1}), \boldsymbol{z}(t_n), \Delta t] \tag{5.5}$$

for any quantity $\boldsymbol{f}$ dependent on $\boldsymbol{z}(t_{n+1})$, $\boldsymbol{z}(t_n)$ and Δt, (5.4) becomes

$$\boldsymbol{g}[\boldsymbol{z}(t_{n+1})] = \widehat{\boldsymbol{B}}(\boldsymbol{\zeta})\boldsymbol{z}_n - \mathscr{D}(\boldsymbol{\zeta})\boldsymbol{z}(t_{n+1}). \tag{5.6}$$

We now assume that $\boldsymbol{z}(t)$ is analytic in a neighbourhood of t_n, and that β, γ and ξ are also analytic functions of t within the same neighbourhood. A Taylor series expansion for $\boldsymbol{z}(t_{n+1})$ about $t = t_n$ then gives

$$\boldsymbol{z}(t_{n+1}) = \boldsymbol{z}(t_n + \Delta t) = \sum_{s=0}^{\infty} \frac{\boldsymbol{z}^{(s)}(t_n)}{s!}\Delta t^s, \tag{5.7}$$

where $\boldsymbol{z}^{(s)} \equiv \dfrac{\mathrm{d}^s}{\mathrm{d}t^s}\{\boldsymbol{z}\}$; this leads to the series expansions

$$\widehat{\boldsymbol{B}}(\boldsymbol{\zeta}) = \sum_{s=0}^{\infty} \widehat{\boldsymbol{B}}_s(t_n)\Delta t^s \quad \text{and} \quad \mathscr{D}(\boldsymbol{\zeta}) = \sum_{s=0}^{\infty} \mathscr{D}_s(t_n)\Delta t^s, \tag{5.8}$$

where the coefficients $\widehat{\boldsymbol{B}}_s$ and $\mathscr{D}_s$ are fully defined at time t_n. Inserting (5.7) and (5.8) into (5.6) gives

$$\boldsymbol{g}[\boldsymbol{z}(t_{n+1})] = \left(\sum_{s=0}^{\infty} \widehat{\boldsymbol{B}}_s \Delta t^s\right)\boldsymbol{z}_n - \left(\sum_{s=0}^{\infty} \mathscr{D}_s \Delta t^s\right)\left(\sum_{s=0}^{\infty} \frac{\boldsymbol{z}^{(s)}(t_n)}{s!}\Delta t^s\right). \tag{5.9}$$

We now use the fact that the exact solution satisfies (2.4), which in matrix form is

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{C}(t)\boldsymbol{z}(t) \quad \text{where } \boldsymbol{C}(t) = \begin{pmatrix} \boldsymbol{0}_3 & \frac{1}{m}\boldsymbol{I}_3 \\ -f(t)\boldsymbol{I}_3 & \boldsymbol{0}_3 \end{pmatrix} \text{ and } f(t) := \frac{\widetilde{V}'[l(t)]}{l(t)} \tag{5.10}$$

with $\boldsymbol{0}_3 \in \mathbb{R}^{3\times 3}$ representing the zero matrix. We can then relate the derivatives $\boldsymbol{z}^{(s)}$ to $\boldsymbol{z}$ itself by repeated differentiation of $(5.10)_1$, i.e.

$$\ddot{\boldsymbol{z}} = \dot{\boldsymbol{C}}\boldsymbol{z} + \boldsymbol{C}\dot{\boldsymbol{z}} = (\dot{\boldsymbol{C}} + \boldsymbol{C}^2)\boldsymbol{z},$$

$$\boldsymbol{z}^{(3)} = \frac{\mathrm{d}}{\mathrm{d}t}\{\dot{\boldsymbol{C}} + \boldsymbol{C}^2\}\boldsymbol{z} + (\dot{\boldsymbol{C}} + \boldsymbol{C}^2)\dot{\boldsymbol{z}} = (\ddot{\boldsymbol{C}} + \boldsymbol{C}\dot{\boldsymbol{C}} + 2\dot{\boldsymbol{C}}\boldsymbol{C} + \boldsymbol{C}^3)\boldsymbol{z}, \ldots$$

and so on (note that $\boldsymbol{C}$ and $\dot{\boldsymbol{C}}$ do not commute). Summarising this procedure, we have

$$\frac{\boldsymbol{z}^{(s)}}{s!} = \boldsymbol{C}_s\boldsymbol{z};\ \ s \geqslant 0, \quad \text{where } \boldsymbol{C}_0 = \boldsymbol{I}_6 \text{ and } \boldsymbol{C}_{s+1} = \frac{1}{s+1}(\dot{\boldsymbol{C}}_s + \boldsymbol{C}_s\boldsymbol{C}) \text{ for } s \geqslant 0. \tag{5.11}$$

Substituting (5.11) into (5.9) results in

$$\begin{aligned}\boldsymbol{g}[\boldsymbol{z}(t_{n+1})] &= \left(\sum_{s=0}^{\infty} \widehat{\boldsymbol{B}}_s \Delta t^s\right)\boldsymbol{z}_n - \left(\sum_{s=0}^{\infty} \mathscr{D}_s \Delta t^s\right)\left(\sum_{s=0}^{\infty} \boldsymbol{C}_s \Delta t^s\right)\boldsymbol{z}_n \\ &= \left[\sum_{s=0}^{\infty}\left(\widehat{\boldsymbol{B}}_s - \sum_{r=0}^{s} \mathscr{D}_r \boldsymbol{C}_{s-r}\right)\Delta t^s\right]\boldsymbol{z}_n. \end{aligned} \tag{5.12}$$

Given that $\boldsymbol{z}_n \in \mathscr{O}(1)$ [2] i.e. $\mathscr{O}(\Delta t^0)$, we can say that

$$\boldsymbol{g}[\boldsymbol{z}(t_{n+1})] \in \mathscr{O}(\Delta t^{p+1}) \iff \widehat{\boldsymbol{B}}_s = \sum_{r=0}^{s} \mathscr{D}_r \boldsymbol{C}_{s-r} \text{ for } s = 0, \ldots, p,$$

i.e. the matrix multiplying $\boldsymbol{z}_n$ in (5.12) must be zero at each power of Δt up to Δt^p. Now, from (5.3) this is equivalent to

$$\nabla\boldsymbol{g}(\boldsymbol{z}_{n+1})\boldsymbol{\epsilon} + \mathscr{O}(\|\boldsymbol{\epsilon}\|^2) \in \mathscr{O}(\Delta t^{p+1}) \iff \widehat{\boldsymbol{B}}_s = \sum_{r=0}^{s} \mathscr{D}_r \boldsymbol{C}_{s-r} \text{ for } s = 0, \ldots, p.$$

Since $\boldsymbol{g}(\boldsymbol{x}) \in \mathscr{O}(1)$ for general $\boldsymbol{x}$ provided $\mathscr{D}_0 \neq 0$, the matrix $\nabla\boldsymbol{g}(\boldsymbol{z}_{n+1}) \in \mathscr{O}(1)$ also, and so for $\mathscr{D}_0 \neq 0$ we have

$$\boldsymbol{\epsilon} \in \mathscr{O}(\Delta t^{p+1}) \iff \widehat{\boldsymbol{B}}_s = \sum_{r=0}^{s} \mathscr{D}_r \boldsymbol{C}_{s-r} \text{ for } s = 0, \ldots, p. \tag{5.13}$$

The integer p is known as the *order of accuracy* for Algorithm 1. The higher the value of p, the more rapidly the local error decreases to zero as $\Delta t \to 0$. Thus we seek to find schemes with p as high as possible, in the hope that this provides us with acceptably accurate solutions for practical values of Δt.

[2] $\mathscr{O}(\Delta t^s)$ denotes the set of functions $\{\boldsymbol{f}(\Delta t) : \|\boldsymbol{f}(\Delta t)\| \leqslant a_f|\Delta t^s|\ \forall \Delta t \leqslant \Delta t_{\mathrm{cr}}\}$, where a_f is a scalar (determined by $\boldsymbol{f}$) which is *independent* of Δt, and $\Delta t_{\mathrm{cr}} > 0$. It can also denote a member of this set.

Note: The concept of order of accuracy of an algorithm relates to the *local* error, rather than the global error, as seen from (5.13). Thus a pth-order algorithm guarantees that the local error is $\mathcal{O}(\Delta t^{p+1})$. If, in addition, the algorithm is known to be *stable* for some range $\Delta t \in [0, \Delta t_{\text{cr}}]$, then the global error will be $\mathcal{O}(\Delta t^{p})$ for $\Delta t \leqslant \Delta t_{\text{cr}}$. Thus the order of accuracy also denotes the *exponent of the global error for a stable solution*.

We now establish the criteria for Algorithm 1 to be pth-order accurate in terms of the parameters β, γ and ξ. First, we express the coefficient matrices $\widehat{\boldsymbol{B}}_s$ and $\boldsymbol{C}_s$ as

$$\widehat{\boldsymbol{B}}_s := \begin{pmatrix} \widehat{B}_{11,s}\boldsymbol{I}_3 & \widehat{B}_{12,s}\boldsymbol{I}_3 \\ \widehat{B}_{21,s}\boldsymbol{I}_3 & \widehat{B}_{22,s}\boldsymbol{I}_3 \end{pmatrix} \quad \text{and} \quad \boldsymbol{C}_s := \begin{pmatrix} C_{11,s}\boldsymbol{I}_3 & C_{12,s}\boldsymbol{I}_3 \\ C_{21,s}\boldsymbol{I}_3 & C_{22,s}\boldsymbol{I}_3 \end{pmatrix}$$

so that from (3.9) and $(5.8)_1$ we have $\widehat{\boldsymbol{B}}_{ij} = \sum_{s=0}^{\infty} \widehat{\boldsymbol{B}}_{ij,s}\Delta t^s$ for $i,j = 1,2$. Then for $p = 0, 1, 2, \ldots$, (5.13) gives us

$$\begin{aligned} &\text{0th order:} \quad \widehat{\boldsymbol{B}}_{ij,0} = \mathscr{D}_0 C_{ij,0} \quad \text{where } \mathscr{D}_0 \neq 0, \\ &\text{1st order:} \quad \text{0th order} \quad \text{and} \quad \widehat{\boldsymbol{B}}_{ij,1} = \mathscr{D}_0 C_{ij,1} + \mathscr{D}_1 C_{ij,0}, \\ &\text{2nd order:} \quad \text{1st order} \quad \text{and} \quad \widehat{\boldsymbol{B}}_{ij,2} = \mathscr{D}_0 C_{ij,2} + \mathscr{D}_1 C_{ij,1} + \mathscr{D}_2 C_{ij,0}, \\ &\langle \text{etc.} \rangle \end{aligned} \tag{5.14}$$

for $i,j = 1,2$. From (3.8) and (3.9) we have the relations

$$\begin{aligned} &\widehat{\boldsymbol{B}}_{11} = \left(\beta + \frac{1}{2}\gamma\right)^2 - \frac{1}{4m}\xi\Delta t^2, \quad \widehat{\boldsymbol{B}}_{12} = \frac{1}{m}\beta\Delta t, \quad \widehat{\boldsymbol{B}}_{21} = -\xi\beta\Delta t, \\ &\widehat{\boldsymbol{B}}_{22} = \left(\beta - \frac{1}{2}\gamma\right)^2 - \frac{1}{4m}\xi\Delta t^2 \quad \text{and} \quad \mathscr{D} = \beta^2 - \frac{1}{4}\gamma^2 + \frac{1}{4m}\xi\Delta t^2 \neq 0, \end{aligned} \tag{5.15}$$

and we can express $\beta(\boldsymbol{\zeta})$, $\gamma(\boldsymbol{\zeta})$ and $\xi(\boldsymbol{\zeta})$ as

$$\beta(\boldsymbol{\zeta}) = \sum_{s=0}^{\infty} \beta_s \Delta t^s, \quad \gamma(\boldsymbol{\zeta}) = \sum_{s=0}^{\infty} \gamma_s \Delta t^s \quad \text{and} \quad \xi(\boldsymbol{\zeta}) = \sum_{s=0}^{\infty} \xi_s \Delta t^s, \tag{5.16}$$

where the coefficients β_s, γ_s and ξ_s are defined at time t_n. From (5.14), the task now amounts to expressing the $\widehat{\boldsymbol{B}}_{ij,s}$ in terms of β_s etc. (where $0 \leqslant s \leqslant p$) by gathering together coefficients of Δt^p on the right-hand sides of (5.15). We must then equate them with the quantities on the right-hand sides of (5.14), of which the $C_{ij,s-r}$ $(0 \leqslant r \leqslant s)$ are obtained by repeated differentiation of the equations of motion, as shown in (5.11). Even for small values of p, this becomes quite involved, so some algebraic steps will not be fully elaborated in our derivations below.

5.1.1. Zeroth-order accuracy

For $p = 0$, the conditions are

$$\mathscr{D}_0 \neq 0 \quad \text{and} \quad \widehat{\boldsymbol{B}}_{ij,0} = \mathscr{D}_0 C_{ij,0}, \;\; i,j = 1,2, \tag{5.17}$$

where, using (5.15) and (5.16), we have

$$B_{11,0} = \left(\beta_0 + \frac{1}{2}\gamma_0\right)^2, \quad B_{12,0} = B_{21,0} = 0, \quad B_{22,0} = \left(\beta_0 - \frac{1}{2}\gamma_0\right)^2 \quad \text{and} \quad \mathscr{D}_0 = \beta_0^2 - \frac{1}{4}\gamma_0^2,$$

and, since $\boldsymbol{C}_0 = \boldsymbol{I}_6$ from (5.11),

$$C_{11,0} = 1, \quad C_{12,0} = C_{21,0} = 0, \quad \text{and} \quad C_{22,0} = 1.$$

Condition (5.17) then reduces to $B_{11,0} = B_{22,0} = \mathscr{D}_0 \neq 0$, thus

$$\left(\beta_0 + \frac{1}{2}\gamma_0\right) = \left(\beta_0 - \frac{1}{2}\gamma_0\right) \neq 0.$$

Therefore we have zeroth-order accuracy if and only if

$$\gamma_0 = 0 \quad \text{and} \quad \beta_0 \neq 0. \tag{5.18}$$

This property assures that the *local* error will tend to zero as $\Delta t \to 0$. Note that by itself, this is *not* sufficient to assure convergence.

5.1.2. First-order accuracy

For $p = 1$, the conditions include those for $p = 0$ and also

$$\widehat{\boldsymbol{B}}_{ij,1} = \mathscr{D}_0 C_{ij,1} + \mathscr{D}_1 C_{ij,0}, \quad i,j = 1,2, \tag{5.19}$$

where, using (5.16), (5.15) and (5.18), we have

$$B_{11,1} = 2\beta_0\left(\beta_1 + \frac{1}{2}\gamma_1\right), \quad B_{12,1} = \frac{1}{m}\beta_0, \quad B_{21,1} = -\xi_0\beta_0,$$
$$B_{22,1} = 2\beta_0\left(\beta_1 - \frac{1}{2}\gamma_1\right), \quad \mathscr{D}_0 = \beta_0^2 \quad \text{and} \quad \mathscr{D}_1 = 2\beta_0\beta_1.$$

From (5.10) and (5.11) we have

$$\boldsymbol{C}_1 = \begin{pmatrix} \boldsymbol{0}_3 & \frac{1}{m}\boldsymbol{I}_3 \\ -f(t_n)\boldsymbol{I}_3 & \boldsymbol{0}_3 \end{pmatrix}$$

and so

$$C_{11,1} = 0, \quad C_{12,1} = \frac{1}{m}, \quad C_{21,1} = -f(t_n), \quad \text{and} \quad C_{22,1} = 0.$$

Condition (5.19) then reduces to

$$2\beta_0\left(\beta_1 + \frac{1}{2}\gamma_1\right) = 2\beta_0\left(\beta_1 - \frac{1}{2}\gamma_1\right) = 2\beta_0\beta_1, \quad \frac{1}{m}\beta_0 = \beta_0^2\frac{1}{m} \quad \text{and} \quad -\xi_0\beta_0 = -\beta_0^2 f(t_n)$$

which leads to

$$\gamma_1 = 0, \quad \beta_0 = 1 \quad \text{and} \quad \xi_0 = f(t_n). \tag{5.20}$$

This property is known as *consistency*: for stable algorithms, it assures that the *global* error will tend to zero as $\Delta t \to 0$, and thus implies convergence.

5.1.3. Second-order accuracy

For $p = 2$, the conditions include those for $p = 1$ and also

$$\widehat{\boldsymbol{B}}_{ij,2} = \mathscr{D}_0 C_{ij,2} + \mathscr{D}_1 C_{ij,1} + \mathscr{D}_2 C_{ij,0}, \quad i,j = 1,2, \tag{5.21}$$

where, using (5.15), (5.16), (5.18) and (5.20), we have

$$B_{11,2} = \beta_1^2 + 2\beta_2 + \gamma_2 - \frac{1}{4m}f(t_n), \quad B_{12,2} = \frac{1}{m}\beta_1, \quad B_{21,2} = -[f(t_n)\beta_1 + \xi_1], \quad \mathscr{D}_0 = 1,$$
$$B_{22,2} = \beta_1^2 + 2\beta_2 - \gamma_2 - \frac{1}{4m}f(t_n), \quad \mathscr{D}_1 = 2\beta_1 \quad \text{and} \quad \mathscr{D}_2 = \beta_1^2 + 2\beta_2 + \frac{1}{4m}f(t_n).$$

From (5.10) and (5.11) we have

$$\boldsymbol{C}_2 = \frac{1}{2}\begin{pmatrix} -\frac{1}{m}f(t_n)\boldsymbol{I}_3 & \boldsymbol{0}_3 \\ -\dot{f}(t_n)\boldsymbol{I}_3 & -\frac{1}{m}f(t_n)\boldsymbol{I}_3 \end{pmatrix}$$

and so

$$C_{11,2} = -\frac{1}{2m}f(t_n), \quad C_{12,2} = 0, \quad C_{21,2} = -\frac{1}{2}\dot{f}(t_n) \quad \text{and} \quad C_{22,2} = -\frac{1}{2m}f(t_n).$$

Condition (5.21) then reduces to

$$\beta_1^2 + 2\beta_2 + \gamma_2 - \frac{1}{4m}f(t_n) = \beta_1^2 + 2\beta_2 - \gamma_2 - \frac{1}{4m}f(t_n) = \beta_1^2 + 2\beta_2 - \frac{1}{4m}f(t_n),$$
$$\frac{1}{m}\beta_1 = 2\beta_1\frac{1}{m} \quad \text{and} \quad -[f(t_n)\beta_1 + \xi_1] = -\frac{1}{2}\dot{f}(t_n) - 2\beta_1 f(t_n)$$

which leads to

$$\gamma_2 = 0, \quad \beta_1 = 0 \quad \text{and} \quad \xi_1 = \tfrac{1}{2}\dot{f}(t_n). \tag{5.22}$$

Continuing in this manner, we derive the criteria for accuracy which are given in Table 1 up to sixth order.

Two important conclusions drawn from this table are as follows:

(1) *The capacity for higher-order local accuracy of Algorithm 1 appears to be limitless*. For each order of accuracy, three extra conditions are necessary (and sufficient when combined with all previous conditions from lower orders). Each of these conditions introduces a *new* coefficient (of β, γ or ξ) on the left-hand side of the equation, which is therefore unassigned: hence no conflict can arise with any previous conditions. Using the symbolic computation package Maple [28], we have verified that this continues to be true for all $p \leqslant 14$. As can be seen from Table 1, however, the complexity of the expressions increases with the accuracy.

(2) *For general potential functions $\widetilde{V}(l)$, the limit for time-integration schemes with constant $\beta = \beta_0$ or $\gamma = \gamma_0$ is second-order accuracy*. If β is held constant, Table 1 shows that we must have $\beta = 1$, in which case $f_n = 0$ is required for third-order accuracy. Similarly, if γ is to be constant, then it must be zero; third-order accuracy then requires that $\dot{f}_n = 0$.

Table 1
Cumulative conditions for pth-order accuracy, where $f_n^{(s)} \equiv \dfrac{\mathrm{d}^s}{\mathrm{d}t^s}\left\{\dfrac{\tilde{V}'[l(t)]}{l(t)}\right\}_{t=t_n}$

p	Conditions needed
0	$\beta_0 \neq 0,\ \gamma_0 = 0$
1	$\beta_0 = 1,\ \xi_0 = f_n,\ \gamma_1 = 0$
2	$\beta_1 = 0\ ,\ \xi_1 = \dfrac{1}{2}\dot{f}_n,\ \gamma_2 = 0$
3	$\beta_2 = -\dfrac{1}{12m}f_n,\ \xi_2 = \dfrac{1}{6}\ddot{f}_n,\ \gamma_3 = \dfrac{1}{12m}\dot{f}_n$
4	$\beta_3 = -\dfrac{1}{24m}\dot{f}_n,\ \xi_3 = \dfrac{1}{24}f_n^{(3)},\ \gamma_4 = \dfrac{1}{24m}\ddot{f}_n$
5	$\beta_4 = -\dfrac{1}{720m^2}(12m\ddot{f}_n + f_n^2),\ \xi_4 = \dfrac{1}{120}f_n^{(4)} + \dfrac{1}{120m}\dot{f}_n^2 - \dfrac{1}{180m}f_n\ddot{f}_n,\ \gamma_5 = \dfrac{1}{720m^2}(9mf_n^{(3)} + 4f_n\dot{f}_n)$
6	$\beta_5 = -\dfrac{1}{1440m^2}(7mf_n^{(3)} + 2f_n\dot{f}_n),\ \xi_5 = \dfrac{1}{720}f_n^{(5)} + \dfrac{1}{180m}\dot{f}_n\ddot{f}_n - \dfrac{1}{360m}f_nf_n^{(3)},$
	$\gamma_6 = \dfrac{1}{360m^2}(mf_n^{(4)} + f_n\ddot{f}_n + \dot{f}_n^2)$

Remark 3. If α, β and ξ are chosen as the free parameters, as in (3.6), the resulting family of algorithms can be *at most second-order accurate* for general potential functions [27].

5.2. Connection between conservation properties and local accuracy

At this point it is natural to ask whether or not any of the accuracy requirements given in Table 1 conflict with the conservation conditions given in Section 4. Given that the exact solution possesses all of the conservation properties we have mentioned, we would expect there to be no conflict, so that higher-order accuracy is not limited by any of the conservation criteria. This is indeed the case, and can be shown very simply as follows. We define the *local energy error* ε^H, the *local angular momentum error* $\boldsymbol{\varepsilon}^{\mathscr{J}}$ and the *local relative equilibrium error* $\boldsymbol{\varepsilon}^R$ as

$$\begin{aligned}
&\varepsilon^H = H_{n+1} - H(t_{n+1}),\\
&\boldsymbol{\varepsilon}^{\mathscr{J}} = \mathscr{J}_{n+1} - \mathscr{J}(t_{n+1}) \quad \text{and}\\
&\boldsymbol{\varepsilon}^R = [\, \|\boldsymbol{q}_{n+1}\| - \|\boldsymbol{q}(t_{n+1})\| \quad \|\boldsymbol{p}_{n+1}\| - \|\boldsymbol{p}(t_{n+1})\| \quad \boldsymbol{q}_{n+1}\cdot\boldsymbol{p}_{n+1} - \boldsymbol{q}(t_{n+1})\cdot\boldsymbol{p}(t_{n+1}) \,]^{\mathrm{T}}
\end{aligned} \tag{5.23}$$

when $\boldsymbol{z}_n = \boldsymbol{z}(t_n)$, in keeping with the local error vector $\boldsymbol{\epsilon}$. Note that in contrast to ε^H and $\boldsymbol{\varepsilon}^{\mathscr{J}}$, $\boldsymbol{\varepsilon}^R$ is valid only when the initial conditions are as given in (4.4).

Theorem 1. *Any pth-order algorithm conserves energy and angular momentum and also preserves the orbits of relative equilibria up to order p or higher. That is to say*

$$\boldsymbol{\epsilon} \in \mathscr{O}(\Delta t^{p+1}) \;\Rightarrow\; \varepsilon^H \in \mathscr{O}(\Delta t^{p+p_1}),\ \boldsymbol{\varepsilon}^{\mathscr{J}} \in \mathscr{O}(\Delta t^{p+p_2}) \text{ and } \boldsymbol{\varepsilon}^R \in \mathscr{O}(\Delta t^{p+p_3}),$$

where $p_1, p_2, p_3 \geqslant 1$.

The proof is given in Appendix A.4, and the result was also stated in [1] for energy and momentum conservation in particular. An important corollary of Theorem 1 is the following:

Corollary 1. *Any algorithm with a local energy error* ε^H *of order* $p+1$ *can be at most pth-order accurate.*

Therefore algorithms that are designed to *dissipate* energy will have an *upper limit on their order of accuracy* prescribed by the amount of energy dissipated at each time-step. For example, the energy-decaying algorithm of Bauchau and Joo [20], derived from a time-discontinuous Galerkin approximation of system (2.4), is a dissipative scheme which is shown empirically to be third-order accurate. From Theorem 1, we conclude that the amount of energy dissipated by this algorithm within each time-step is $\mathscr{O}(\Delta t^4)$ or smaller.

It is also natural to ask whether fulfilment of the conservation conditions alone requires a certain order of accuracy. Interestingly, the answer is no; in fact, there exist fully conserving algorithms that do not fulfil *any* of the criteria given in Table 1. For example, the scheme

$$\begin{aligned}
\boldsymbol{q}_\Delta &= \frac{1}{\xi \Delta t}\boldsymbol{p}_{1/2},\\
\boldsymbol{p}_\Delta &= -\frac{m}{\Delta t}\boldsymbol{q}_{1/2}
\end{aligned}$$

obtained from Algorithm 1 using $\beta = \frac{1}{m}\Delta t^2 \xi$ and $\gamma = 0$ not only conserves angular momentum, but also conserves energy, preserves the orbits of relative equilibria and is time-reversible when $\xi = \widetilde{V}_\Delta / [\frac{1}{2}(l_{n+1}^2 - l_n^2)]$, as seen from (4.2), (4.8) and (4.16). Since $\beta_0 = 0$, however, the very first condition in Table 1 is violated. We see that as $\Delta t \to 0$ we have $\boldsymbol{q}_{n+1} \to -\boldsymbol{q}_n$ and $\boldsymbol{p}_{n+1} \to -\boldsymbol{p}_n$, so the scheme is clearly not convergent. A similar observation in relation to angular momentum conservation for algorithms dealing with rotating rigid bodies was made in [29].

In certain circumstances, however, we can make a converse statement to Theorem 1 with regard to energy, as given in the following theorem:

Theorem 2. *If β and γ fulfil the conditions in Table 1 for pth-order accuracy where $p \geqslant 1$, then ξ chosen to conserve energy up to order p or higher will assure that the algorithm is pth-order accurate.*

The proof is given in Appendix A.5. For algorithms that conserve energy, then, the accuracy characteristics are dictated entirely by β and γ: thus we can remove ξ from consideration when we know that our algorithm is energy-conserving. This choice of ξ is therefore *optimal* for accuracy.

6. Links to existing algorithms

In this section, we show how some of the existing algorithms are related to our work on central-force problems here.

6.1. Individual momentum-conserving algorithms

The following algorithms are all instances of Algorithm 1 with parameters $\beta = 1$ and $\gamma = 0$, and each one is second-order accurate and time-reversible:

- the symplectic-momentum mid-point rule (see [4,30–32] amongst many others), referred to here as SMM, with $\xi = \frac{\tilde{V}'(\|\boldsymbol{q}_{1/2}\|)}{\|\boldsymbol{q}_{1/2}\|}$;
- the energy–momentum mid-point algorithm [2–8], referred to here as EMM, with $\xi = \frac{\tilde{V}_\Delta}{\frac{1}{2}(l_{n+1}^2 - l_n^2)}$ as given in (4.3); and
- the "assumed distance method" [18,33], referred to here as ADM, with $\xi = \frac{\tilde{V}'(l_{1/2})}{l_{1/2}}$.

Note that both EMM and ADM preserve the orbits of relative equilibria, whereas SMM does not. Also, ADM is not energy-conserving in general, although it becomes so for quadratic potential functions, whereupon it coincides with EMM [34].

6.2. Families of momentum-conserving algorithms

As mentioned in Section 3.2, Algorithm 1 follows on from the three-parameter family (3.6) presented in [24], which itself was a generalisation of EMM. An earlier three-parameter family is that of Simo et al. given in Eq. (2.19) of [4]; for a central-force problem, this family is equivalent to (3.6) with

$$\beta = \frac{1}{\kappa_1} \quad \text{and} \quad \xi = \frac{\kappa_2 \widetilde{V}'(\|\boldsymbol{q}_\alpha\|)}{\kappa_1 \|\boldsymbol{q}_\alpha\|}.$$

Thus the algorithms will conserve energy if (4.2) is satisfied, which corresponds exactly to Eq. (2.22) of [4] in terms of α, κ_1 and κ_2.

Recently, Armero and Romero [9] proposed a family of algorithms that dissipate energy in the non-linear regime for potentials $\widetilde{V}(l)$ such that $\widetilde{V}''(l) \geqslant 0 \ \forall l$. It coincides with Algorithm 1 if the choices $\gamma = 0$,

$$\beta = \frac{\|\boldsymbol{p}_n\| + \|\boldsymbol{p}_{n+1}\|}{(1-\chi_2)\|\boldsymbol{p}_n\| + (1+\chi_2)\|\boldsymbol{p}_{n+1}\|} \quad \text{and} \quad \xi = \beta \frac{\widetilde{V}_\Delta + 4\chi_1[\widetilde{V}_{1/2} - \widetilde{V}(l_{1/2})]}{\frac{1}{2}(l_{n+1}^2 - l_n^2)}$$

are made, where χ_1 and χ_2 are constant parameters used to control the amount of energy dissipated. This family, named EDMC-1, is given in Eq. (2.28) of [9], and extends the earlier dissipative schemes of Armero

and Petőcz [35], which are recovered when $\chi_2 = 0$. If $\chi_1 = \chi_2 = 0$, or $l_{n+1} = l_n$ and $\|\boldsymbol{p}_{n+1}\| = \|\boldsymbol{p}_n\|$, EMM is recovered, hence this family preserves relative equilibria for general χ_1 and χ_2. The algorithm is first-order accurate unless $\chi_1 = \chi_2 = 0$ [9].

A remarkable family of higher-order accurate conservative algorithms was presented by LaBudde and Greenspan in Section 4B of [1] in 1976. They used a *predictor–corrector* formulation

$$\begin{aligned} \boldsymbol{q}_{n+1} &= \hat{\boldsymbol{q}}_n + \boldsymbol{\delta q}_{n+1}, \\ \boldsymbol{p}_{n+1} &= \hat{\boldsymbol{p}}_n + \boldsymbol{\delta p}_{n+1}, \end{aligned} \tag{6.1}$$

where quantities $\hat{\boldsymbol{q}}_n$ and $\hat{\boldsymbol{p}}_n$ (which are fully determined at time-step n) are the predictors for the solution $\{\boldsymbol{q}_{n+1}, \boldsymbol{p}_{n+1}\}$, and $\boldsymbol{\delta q}_{n+1}$ and $\boldsymbol{\delta p}_{n+1}$ (which require information at time-step $n+1$) the correctors. The predictors were based on Taylor series approximations to the true solution at time t_{n+1}, with the position predictors accurate to one order higher than the momentum predictors: for example,

$$\begin{aligned} \hat{\boldsymbol{q}}_n &:= \boldsymbol{q}_n + \frac{1}{m}\boldsymbol{p}_n \Delta t, \\ \hat{\boldsymbol{p}}_n &:= \boldsymbol{p}_n \end{aligned} \tag{6.2}$$

are the first-order predictors, where the local errors are $\hat{\boldsymbol{q}}_n - \boldsymbol{q}(t_{n+1}) \in \mathcal{O}(\Delta t^2)$ and $\hat{\boldsymbol{p}}_n - \boldsymbol{p}(t_{n+1}) \in \mathcal{O}(\Delta t)$ for $\boldsymbol{q}_n = \boldsymbol{q}(t_n)$ and $\boldsymbol{p}_n = \boldsymbol{p}(t_n)$. The correctors $\boldsymbol{\delta q}_{n+1}$ and $\boldsymbol{\delta p}_{n+1}$ are then chosen to provide energy and momentum conservation, and also to secure second-order accuracy [3] in both $\boldsymbol{q}$ and $\boldsymbol{p}$, so that $\boldsymbol{q}_{n+1} - \boldsymbol{q}(t_{n+1}) \in \mathcal{O}(\Delta t^3)$ and $\boldsymbol{p}_{n+1} - \boldsymbol{p}(t_{n+1}) \in \mathcal{O}(\Delta t^3)$. Similarly,

$$\begin{aligned} \hat{\boldsymbol{q}}_n &:= \boldsymbol{q}_n + \frac{1}{m}\boldsymbol{p}_n \Delta t - \frac{1}{2m}\frac{\widetilde{V}'(l_n)}{l_n}\boldsymbol{q}_n \Delta t^2, \\ \hat{\boldsymbol{p}}_n &:= \boldsymbol{p}_n - \frac{\widetilde{V}'(l_n)}{l_n}\boldsymbol{q}_n \Delta t \end{aligned}$$

are the second-order predictors, where $\hat{\boldsymbol{q}}_n - \boldsymbol{q}(t_{n+1}) \in \mathcal{O}(\Delta t^3)$ and $\hat{\boldsymbol{p}}_n - \boldsymbol{p}(t_{n+1}) \in \mathcal{O}(\Delta t^2)$, with $\boldsymbol{\delta q}_{n+1}$ and $\boldsymbol{\delta p}_{n+1}$ now providing energy and momentum conservation and third-order accuracy. The similarities with our work are clear, since this process can obviously be extended to generate energy–momentum schemes of arbitrary order.

It can be shown that the algorithms given by (6.1) are not time-reversible and do not preserve the orbits of relative equilibria if the accuracy is greater than second order. In the case of a second-order algorithm, we have the predictors $\hat{\boldsymbol{q}}_n$ and $\hat{\boldsymbol{p}}_n$ given in (6.2) with correctors

$$\boldsymbol{\delta q}_{n+1} = \frac{\Delta t \epsilon \boldsymbol{a}_n}{2\|\boldsymbol{a}_n\|^2} \quad \text{and} \quad \boldsymbol{\delta p}_{n+1} = \frac{\epsilon \boldsymbol{a}_n}{\|\boldsymbol{a}_n\|^2},$$

where $\boldsymbol{a}_n = \boldsymbol{q}_n + \frac{1}{2m}\Delta t \boldsymbol{p}_n$ (which provides conservation of angular momentum) and ϵ satisfies the equation

$$\epsilon^2 + \frac{2}{m}(\boldsymbol{a}_n \cdot \boldsymbol{p}_n)\epsilon + \frac{2}{m}\|\boldsymbol{a}_n\|^2 \widetilde{V}_\Delta = 0 \tag{6.3}$$

which ensures energy conservation. This corresponds to Algorithm 1 with parameters $\beta = 1$, $\gamma = 0$ and

$$\xi = -\frac{m\epsilon}{\Delta t(\|\boldsymbol{a}_n\|^2 + \frac{1}{4}\Delta t \epsilon)}. \tag{6.4}$$

[3] In [1], the order of accuracy is quoted as being one order higher than that given here, since it is defined as the exponent of the *local* truncation error, rather than the global error for a stable solution.

It is interesting to note that this algorithm differs from EMM only in the way the energy condition is formulated: substituting $\boldsymbol{p}_n$ in terms of $\boldsymbol{q}_n$ and $\boldsymbol{q}_{n+1}$ from (3.1), and also ϵ in terms of ξ from (6.4), into (6.3) would result in (4.2) for $\beta = 1$ and $\gamma = 0$, hence ξ from (4.3) is recovered. Thus these two algorithms give identical results, as was noted in [9], *when a unique solution for* $\boldsymbol{p}_{n+1}$ *and* $\boldsymbol{q}_{n+1}$ *is obtained by each*; the difference lies in the actual non-linear equations that are solved. Expressing $\boldsymbol{q}_{n+1}$ in terms of $\boldsymbol{q}_n$, $\boldsymbol{p}_n$ and ϵ in $\widetilde{V}_\Delta \equiv \widetilde{V}(\|\boldsymbol{q}_{n+1}\|) - \widetilde{V}(\|\boldsymbol{q}_n\|)$ shows that (6.3) can be read as an implicit equation in ϵ, to be solved iteratively, which confines the non-linearity of the algorithm to a single parameter: thus the Newton–Raphson process would involve a scalar equation in ϵ rather than a vector equation in $\boldsymbol{q}_{n+1}$. This benefit does not extend to problems with many degrees of freedom, however, so we will not adopt this approach here.

For completeness, we mention an early family of energy-conserving schemes given by Chorin et al. in Eqs. (3.23)–(3.26) of [2]; for a central-force problem, this becomes

$$\frac{1}{\Delta t}\boldsymbol{q}_\Delta = \frac{1}{m}\left(\frac{\boldsymbol{p}_\Delta \cdot \boldsymbol{p}_{1/2}}{\boldsymbol{p}_\Delta \cdot \boldsymbol{p}_{\bar{\alpha}}}\right)\boldsymbol{p}_{\bar{\alpha}},$$
$$\frac{1}{\Delta t}\boldsymbol{p}_\Delta = -\left(\frac{\widetilde{V}_\Delta}{\boldsymbol{q}_\Delta \cdot \boldsymbol{q}_\alpha}\right)\boldsymbol{q}_\alpha$$

with parameters α and $\bar{\alpha}$. If these are chosen such that $\bar{\alpha} = 1 - \alpha$, then the resultant one-parameter family will also conserve angular momentum, and become equivalent to (3.6) with

$$\beta = \frac{\boldsymbol{p}_\Delta \cdot \boldsymbol{p}_{1-\alpha}}{\boldsymbol{p}_\Delta \cdot \boldsymbol{p}_{1/2}} \quad \text{and} \quad \xi = \beta\frac{\widetilde{V}_\Delta}{\boldsymbol{q}_\Delta \cdot \boldsymbol{q}_\alpha}.$$

7. Description of a model problem

7.1. Problem data

We now wish to test a selection of our algorithms with a model central-force problem that can be made more or less *stiff* by design. We choose the pendulum example shown in Fig. 2, as used by many others [7,9,24,36], with the potential function representative of the St. Venant–Kirchhoff material [5,10,33]

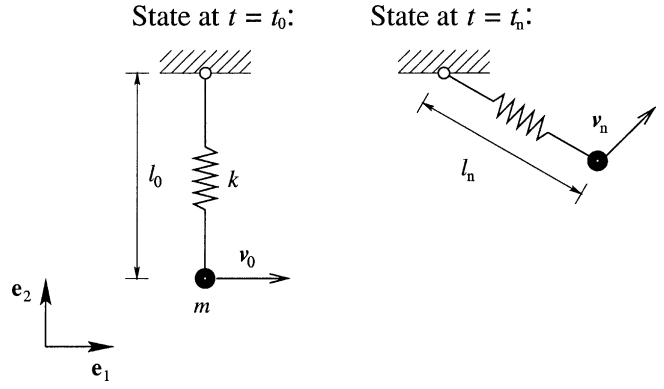

Fig. 2. Pendulum of mass m and stiffness k with initial length l_0 and velocity $\boldsymbol{v}_0$.

$$\widetilde{V}(l) = \frac{1}{2}k\left(\frac{l^2 - \bar{l}^2}{2\bar{l}}\right)^2 \tag{7.1}$$

for constants k and $\bar{l}$, which represent the stiffness and the natural (unstrained) length of the pendulum, respectively. This choice of potential allows the problem to be made arbitrarily stiff by increasing k (while keeping m fixed).

In the tests that follow, we use two different problem definitions. In both cases, the top of the pendulum is fixed at the origin $(0,0)$ with the mass starting at position $\boldsymbol{q}_0^{\mathrm{T}} = [\,0 \quad l_0\,]$ with initial length $l_0 = 1$. The natural length used in the St. Venant–Kirchhoff potential is $\bar{l} = 1$, meaning that there is no initial strain in the pendulum. The mass used is $m = 1$, and the pendulum is fired with an initial horizontal velocity $\boldsymbol{v}_0^{\mathrm{T}} = [\,10 \quad 0\,]$. The two definitions differ only in the choice of stiffness k: we have a non-stiff pendulum with $k = 10^2$ and a stiff pendulum with $k = 10^8$.

7.2. Details of tests

All of the algorithms under consideration will be tested on both of the pendulum problems described in Section 7.1 for a total response time of 0.6 s. Each one will be used with a range of time-step sizes, and the relative errors in the positions $\boldsymbol{q}$ and momenta $\boldsymbol{p}$ will be taken at two sampling times; 0.3 and 0.6 s. The relative errors are calculated as

$$\frac{\|\boldsymbol{q}_n - \boldsymbol{q}(t_n)\|}{\|\boldsymbol{q}(t_n)\|} \quad \text{and} \quad \frac{\|\boldsymbol{p}_n - \boldsymbol{p}(t_n)\|}{\|\boldsymbol{p}(t_n)\|},$$

respectively, where $\{\boldsymbol{q}_n, \boldsymbol{p}_n\}$ denotes the approximate solution and $\{\boldsymbol{q}(t_n), \boldsymbol{p}(t_n)\}$ the reference solution. This reference solution has been obtained by running each of the higher-order schemes several times with decreasing time-step sizes, and using the common solution arising from them, which was consistent up to 15 digits. In order to preserve 15 digits of precision, a quadruple precision module based on the ideas given in [37] has been used in the code. With the aid of quadruple precision arithmetic, we are able to use an extremely tight convergence tolerance for the Newton–Raphson iteration, namely 10^{-30} for the non-stiff pendulum and 10^{-26} for the stiff pendulum; see Appendix B for details. A maximum of 50 Newton–Raphson iterations are carried out before any solution attempt is aborted.

8. Numerical results

8.1. Choice of algorithms

The simplest higher-order algorithms to test are those which have β and γ in (3.7) defined as truncated versions of the power series in Δt given in (5.16), with coefficients matching the requirements for accuracy given in Table 1. Thus β and γ are defined entirely at time-step n, which means that algorithms of this type *cannot* be time-reversible. The remaining parameter ξ is then obtained from (4.2) to ensure energy conservation. The derivatives $f_n^{(s)}$ in Table 1 are now taken as the discrete analogues of the continuous functions $f^{(s)}[l(t_n)] := \frac{\mathrm{d}^s}{\mathrm{d}t^s}\left\{\frac{\tilde{V}'[l(t)]}{l(t)}\right\}_{t=t_n}$, with $\boldsymbol{q}_n$ and $\boldsymbol{p}_n$ replacing $\boldsymbol{q}(t_n)$ and $\boldsymbol{p}(t_n)$: these definitions equate to those used in the higher-order algorithms of LaBudde and Greenspan [1], and ensure that the accuracy criteria are met. We emphasise, however, that our algorithms are designed to preserve relative equilibria: when $\boldsymbol{q}_n \cdot \boldsymbol{p}_n = 0$ and $\frac{1}{m}\|\boldsymbol{p}_n\|^2 = f_n l_n^2$, all derivatives of f_n vanish, resulting in $\gamma^{\mathrm{RE}} = 0$. This in turn implies

$\xi^{\mathrm{RE}} = f_n$ as required by (4.9). We call these algorithms EMp, where p denotes the order of accuracy, and test algorithms EM3, EM4, EM6 and EM10 (which are therefore third-, fourth-, sixth- and tenth-order accurate, respectively); the extra coefficients needed for EM10 are given in [27]. Details of the Newton–Raphson linearisation of these schemes can be found in Appendix B.

For comparison purposes, we will also test an *explicit* version of EM10, named EM10e, where ξ is no longer obtained from (4.2) but instead taken as a power series in Δt along with β and γ. Table 1 shows the coefficients ξ_0–ξ_5 used for ξ; the rest are given in [27]. We note immediately that this scheme is *not* energy-conserving, although it does preserve the orbits of relative equilibria. Since ξ is now also wholly defined at time-step n, the scheme is *linear* with respect to $\boldsymbol{p}_{n+1}$ and $\boldsymbol{q}_{n+1}$; thus there is no Newton–Raphson iteration necessary for the solution.

A more sophisticated approach to designing higher-order algorithms involves finding closed-form expressions for the parameters β and γ which, when expanded in a power series in Δt, match the criteria given in Table 1. These would now involve quantities at time-step $n+1$, thus requiring a more complicated linearisation for the Newton–Raphson procedure. In this way, however, one can design algorithms which are time-reversible. By way of example, we propose the energy-conserving scheme with

$$\beta = \frac{\sqrt{\frac{f_{1/2}}{m}}\frac{\Delta t}{2}}{\tan\left(\sqrt{\frac{f_{1/2}}{m}}\frac{\Delta t}{2}\right)}, \quad \gamma = \frac{\Delta t^2}{12m} f_\Delta \quad \text{and} \quad \xi \text{ given by (4.2)}. \tag{8.1}$$

Expanding β and γ about t_n (with $f(t_{n+1})$ in place of f_{n+1}) and comparing terms with Table 1 shows that the scheme is fourth-order accurate. It is also time-reversible, as seen from (4.16), and satisfies the conditions for preservation of relative equilibria given in (4.8). Following the discussion in Section 4.3, we see that, for this choice of γ, preservation of relative equilibria implies recovery of the *exact* solutions at all time-step sizes for steady-state problems, with β satisfying (4.13). We call this algorithm EMTR4; details of the Newton–Raphson linearisation are given in Appendix B. Given that the trajectory of the stiff pendulum closely resembles that of a steady-state example, we would expect such an algorithm to provide solutions to the stiff problem with minimal error in the angle of rotation.

Note: Eq. (8.1) is by no means the only definition for β and γ that will allow fourth-order accuracy, time reversibility and exact solutions for steady-state problems. Another possibility for β is $\beta = \frac{\sqrt{u}}{\tan\sqrt{u}}$ where

Table 2
Description of the algorithms tested

SMM [4,30–32]	$\beta = 1$, $\gamma = 0$ and $\xi = \dfrac{\tilde{V}'(\|\boldsymbol{q}_{1/2}\|)}{\|\boldsymbol{q}_{1/2}\|}$
EMM [2–8]	$\beta = 1$, $\gamma = 0$ and $\xi = \dfrac{\tilde{V}_\Delta}{\frac{1}{2}(l_{n+1}^2 - l_n^2)}$
ADM [18,33]	$\beta = 1$, $\gamma = 0$ and $\xi = \dfrac{\tilde{V}'(l_{1/2})}{l_{1/2}}$
EMp	$\beta = \sum_{s=0}^{p-1} \beta_s \Delta t^s$, $\gamma = \sum_{s=0}^{p} \gamma_s \Delta t^s$ and $\xi = \xi^*$
EM10e	$\beta = \sum_{s=0}^{9} \beta_s \Delta t^s$, $\gamma = \sum_{s=0}^{10} \gamma_s \Delta t^s$ and $\xi = \sum_{s=0}^{9} \xi_s \Delta t^s$
EMTR4	$\beta = \dfrac{\sqrt{\frac{f_{1/2}}{m}}\frac{\Delta t}{2}}{\tan\left(\sqrt{\frac{f_{1/2}}{m}}\frac{\Delta t}{2}\right)}$, $\gamma = \dfrac{\Delta t^2}{12m} f_\Delta$ and $\xi = \xi^*$

$\xi^* = \dfrac{\beta\tilde{V}_\Delta - \frac{m}{\Delta t^2}\gamma\|\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}\|^2}{(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2})\cdot\boldsymbol{q}_{1/2}}$ and coefficients β_s, γ_s and ξ_s are as given in Table 1 and [27].

$u = \frac{\Delta t^2}{4m} f(l_{1/2})$. Suitable definitions for γ include $\gamma = \frac{1}{2}(\frac{\sqrt{x}}{\tanh\sqrt{x}} - \frac{\sqrt{x}}{\tan\sqrt{x}})$, $1 - \frac{\sqrt{x}}{\tan\sqrt{x}}$, $\frac{1}{3}\tan x$ and $\frac{1}{6}(x + \tan x)$ for $x = \frac{\Delta t^2}{4m} f_\Delta$, among many others.

Finally, for a bench-mark against which to assess the performance of the new algorithms, we will also test algorithms SMM, EMM and ADM as given in Section 6. Table 2 summarises the algorithms tested. As an initial approximation to the solution at time-step $n+1$ with which to begin the iterative process, we use the second-order predictor of LaBudde and Greenspan, namely

$$\boldsymbol{q}_{n+1}^{\langle 0\rangle} := \boldsymbol{q}_n + \frac{\Delta t}{m}\boldsymbol{p}_n - \frac{\Delta t^2}{2m} f_n \boldsymbol{q}_n \tag{8.2}$$

as an alternative to the standard predictor $\boldsymbol{q}_{n+1}^{\langle 0\rangle} := \boldsymbol{q}_n$. Similar alternative predictors were used in [18,33].

Note: We have no way of guaranteeing that the necessary condition $\mathscr{D} \neq 0$ from (3.8) will be satisfied for any of the algorithms used, so we simply halt the analysis if $\mathscr{D}$ becomes less than a tolerance value of 10^{-20}.

8.2. Results

8.2.1. EMp schemes

Figs. 3 and 4 show the relative errors in the positions and momenta for the EMp algorithms tested, with time-step sizes ranging from $\Delta t = 10^{-1}$ to $\Delta t = 10^{-7}$ on both of the example problems, at sampling times $t_n = 0.3$ and 0.6 s. The largest time-step Δt for which an algorithm registers a point on the graph indicates the largest time-step for which Newton–Raphson convergence was achieved. Also, where the lines stop short of the bottom of the graph, the smallest time-step for which a point occurs denotes the last one where a non-zero error was obtained; answers obtained using smaller time-steps matched the reference solution to 15 digits.

From Fig. 3 we see from the slopes of the lines that the theoretical order of accuracy of each algorithm is borne out in practice for the non-stiff pendulum, although the third-order scheme EM3 does not give such regular results as the others. The results are very similar for both sampling times, and in general we see that the errors increase slightly with the sampling time, as expected. We note that the schemes EM3, EM4 and EM6 failed to converge during the Newton–Raphson iteration at $\Delta t = 10^{-1}$. We also see that the explicit scheme EM10e generally gave the lowest errors, notably surpassing the performance of the equivalent energy-conserving scheme EM10. Algorithms SMM, EMM and ADM gave very similar results to one another; this highlights the fact that energy conservation is not essential for this non-stiff problem, as has been similarly demonstrated by other authors (e.g. [18,36]).

For the stiff pendulum, it is clear from Fig. 4 that the theoretical order of accuracy of each algorithm is no longer an indicator of performance, and the results are a little different for the two sampling times. We see that algorithm SMM failed to converge for time-steps $\Delta t \geqslant 10^{-2}$, and all of the new energy-conserving schemes failed to converge for $\Delta t \geqslant 10^{-3}$. We also notice that for any time-step at which an energy-conserving EMp scheme gave a solution, the explicit scheme EM10e gave a (much) more accurate one: whenever the explicit scheme suffered from energy blow-up (namely for $\Delta t \geqslant 10^{-3}$), the conserving schemes failed to provide a converged solution. The third-order scheme EM3 proved the least robust, only registering a solution for the smallest time-step $\Delta t = 10^{-7}$.

Perhaps the most surprising element of these results is the fact that several of the error graphs are not monotonic; thus reducing the time-step size does not guarantee a reduction of error in the solution. This is the case for algorithms EMM and ADM, which gave practically identical results for this problem (as one would expect from [34] given that the trajectories are almost circular, and thus $l_{n+1} \approx l_n \ \forall n$). For these algorithms, a monotonic decrease in the errors did not occur until $\Delta t \leqslant 10^{-5}$, which is approximately $\frac{1}{63}$ of the period of axial vibration for this problem: thus a trend cannot be seen until the time-step is actually small enough to capture the higher mode accurately. For $\Delta t \leqslant 10^{-5}$, algorithm SMM gave indistinguishable

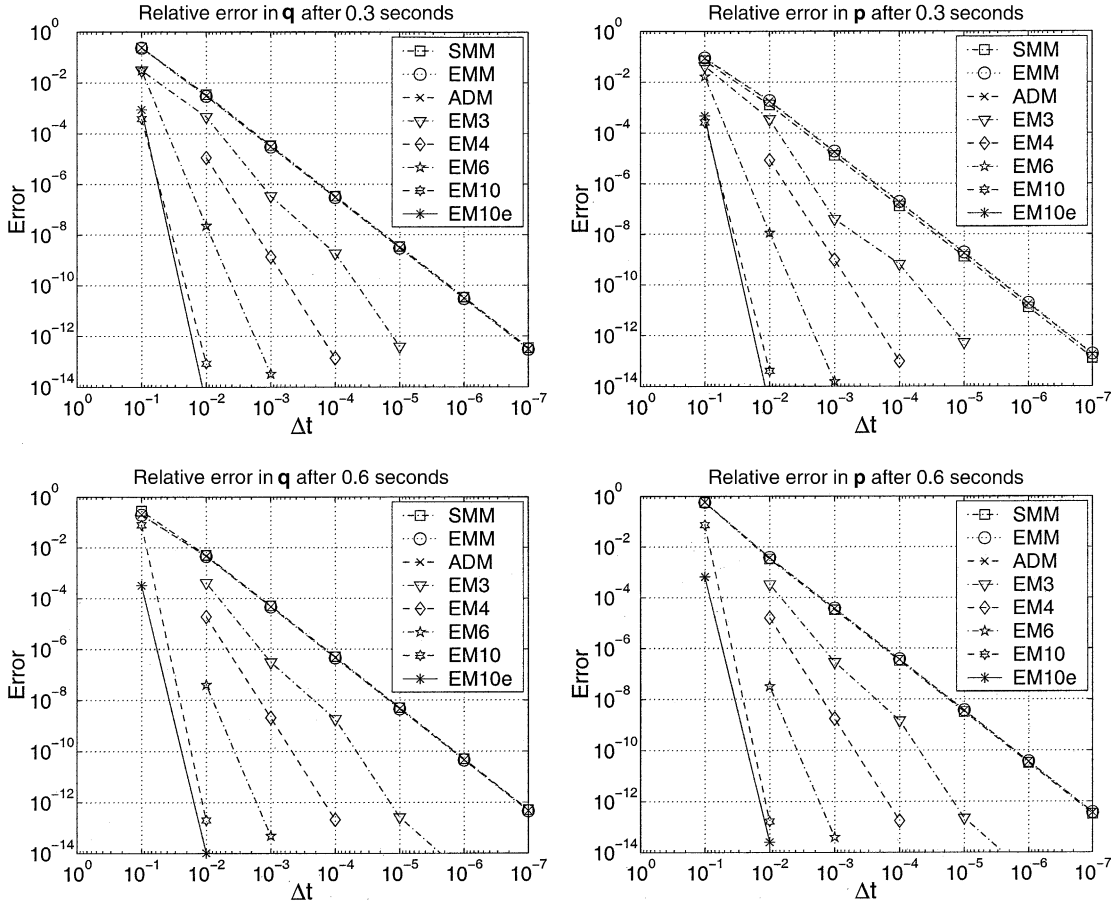

Fig. 3. Relative errors for the non-stiff pendulum.

results to those of EMM and ADM, which agrees with results in [10]. Also, the solutions obtained from schemes EM6 and EM10 were very similar, even for the smallest time-steps; thus the supposed enhanced accuracy of the tenth-order scheme was not apparent in this example. In fact, all of the implicit algorithms that converged gave similar results for $\Delta t = 10^{-4}$. We note that for the established algorithms SMM, EMM and ADM, the graphs become smoother as the sampling time increases, and also that, in contrast to the non-stiff pendulum, the momentum errors here are significantly larger than the position errors in general.

We also ran all of the tests involving the EMp schemes using the standard predictor $\boldsymbol{q}_{n+1}^{\langle 0\rangle} := \boldsymbol{q}_n$ as opposed to that given in (8.2). For the smaller time-step sizes the results were identical, but different results were obtained at some larger time-step sizes for several of the algorithms, including the established algorithm SMM. In all instances, the results obtained with the standard predictor were less accurate than those given in Figs. 3 and 4, although they were often very close.

Concerning the computational cost of these algorithms, we found that the number of Newton–Raphson iterations needed for each was very similar when they converged, as seen in Table 3 for the stiff problem. The increased complexity of the expressions for β, γ and ξ for the higher-order schemes also affected the overall running time, to the extent that algorithms EM10 and EM10e ran approximately 20 and 25 times slower, respectively, than the established schemes.

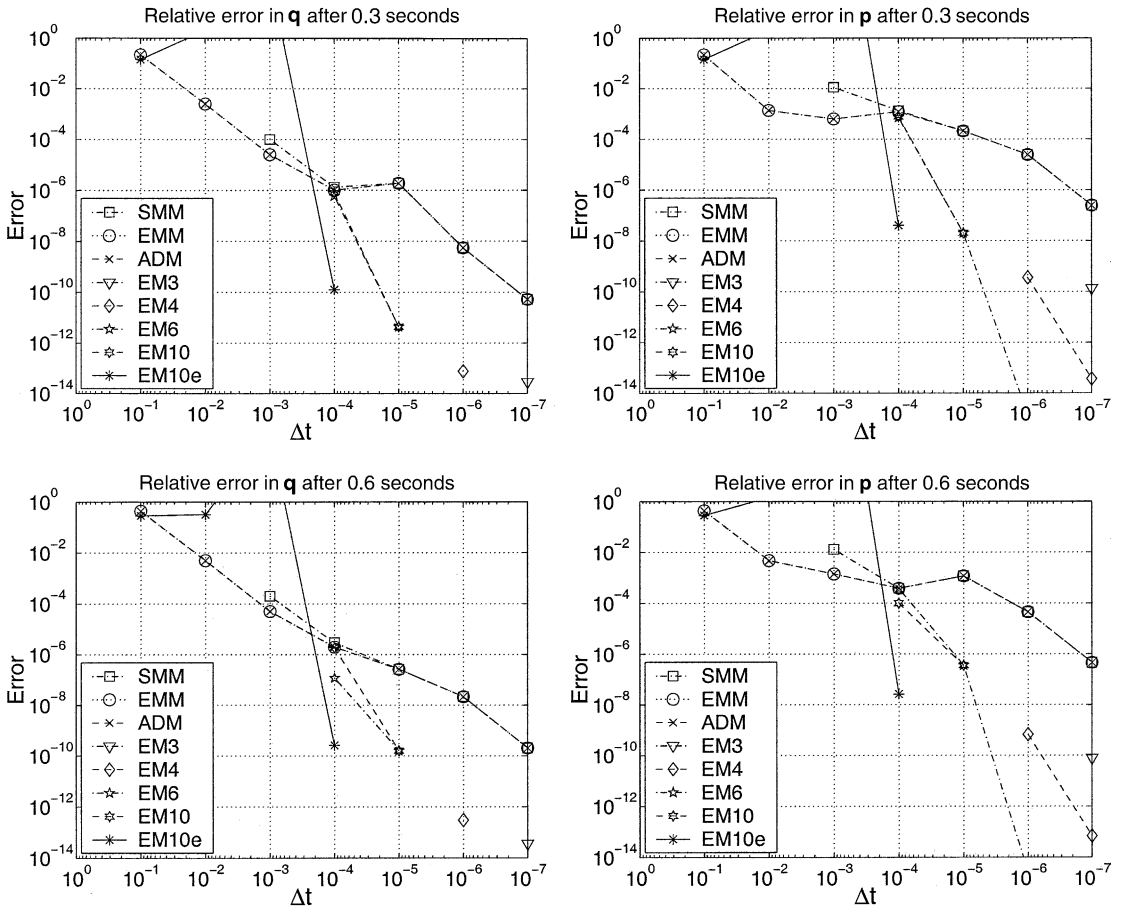

Fig. 4. Relative errors for the stiff pendulum.

Table 3
Average number of Newton–Raphson iterations needed per time-step for convergence for the stiff pendulum

Algorithm	Time-step size Δt						
	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
SMM	*	*	4	2	2	2	1
EMM	9	5	3	2	2	2	1
ADM	8	5	3	2	2	2	1
EM3	*	*	*	*	*	*	1
EM4	*	*	*	*	*	2	1
EM6	*	*	*	4	2	2	–
EM10	*	*	*	4	2	2	–

* Convergence not achieved within 50 iterations.

8.2.2. EMTR4

Figs. 5 and 6 shows results for EMTR4 as compared with algorithms EMM and EM4 for each pendulum at the sampling time $t_n = 0.6$ s, using the second-order predictor given in (8.2). We note the improved

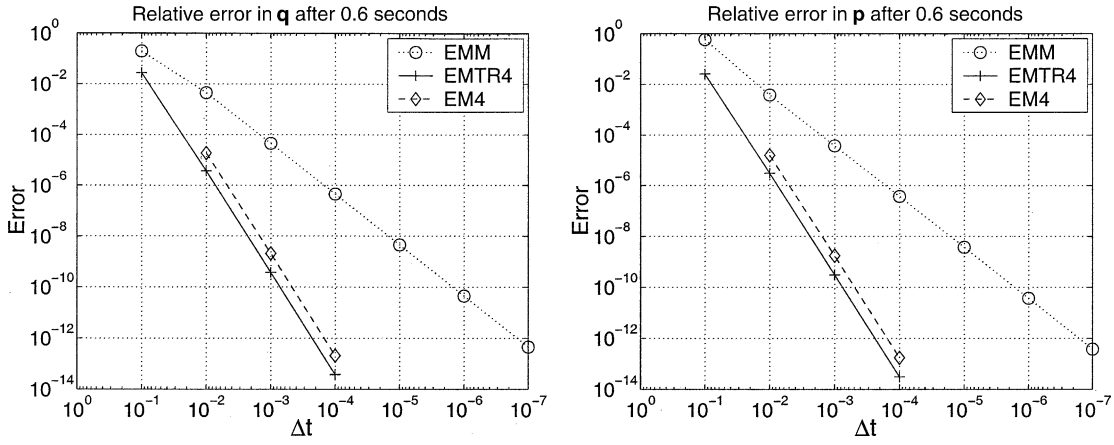

Fig. 5. Relative errors for the non-stiff pendulum.

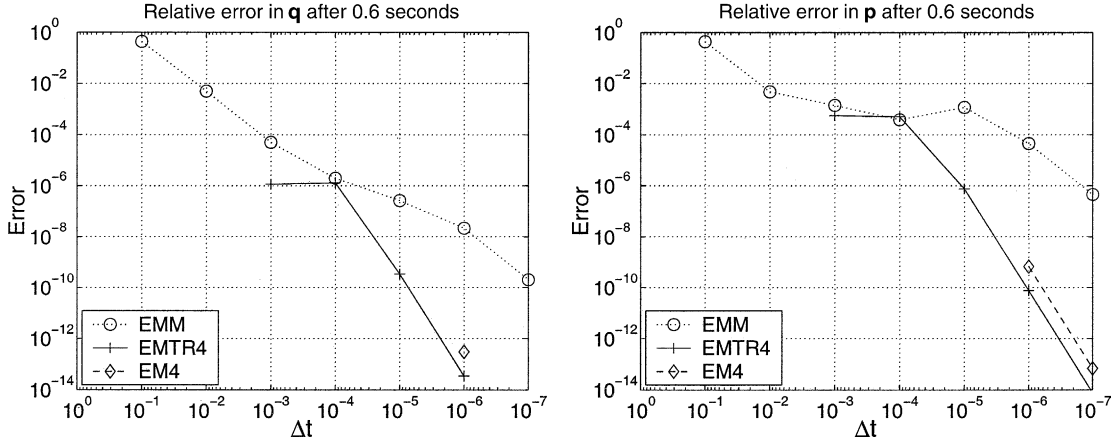

Fig. 6. Relative errors for the stiff pendulum.

performance of EMTR4 over EM4 with respect to Newton–Raphson convergence and the size of the errors for both problems; we also see that the error graph for EMTR4 is monotonic for the stiff pendulum, although not strictly decreasing. However, we note the lack of robustness of EMTR4 when compared to EMM for this problem, with no solution registered for $\Delta t \geqslant 10^{-2}$. For time-steps in this range, the difficulty was not a lack of convergence but rather the algorithm breaking down due to $\mathscr{D} < 10^{-20}$; this in turn was caused by $\beta \ll 1$ occurring during the iteration process. Since $\beta = 1$ is fixed for EMM, this particular difficulty cannot arise, which may explain its superior performance. The largest time-step for which a valid solution to the stiff problem was obtained with EMTR4 was actually $\Delta t = 5 \times 10^{-3}$, or $\frac{1}{70}$ of the period of rotation, and we observe that this scheme only starts to exhibit fourth-order error decay for $\Delta t \leqslant 10^{-4}$, which is roughly $\frac{1}{6}$ of the period of axial vibration. These results were reproduced when using any of the alternative definitions for β and γ mentioned in Section 8.1, although we acknowledge that many more possible definitions exist.

8.3. Discussion

Closer inspection of the construction of β and γ in the EMp schemes reveals extensive use of dot products in the series coefficients β_s and γ_s when forming the time-derivatives $\dot{l}_n$, $\ddot{l}_n$, etc. that are required. Calculations such as these are inevitably subject to round-off error, even when using quadruple precision arithmetic, which can then pervade the rest of the solution. This can be illustrated effectively by considering a steady-state problem, where $\dot{l}_n$ is known to be zero; in practice, we compute $\dot{l}_n$ as $\frac{\boldsymbol{q}_n \cdot \boldsymbol{p}_n}{m l_n}$, and thus we rely on accurately calculating $\boldsymbol{q}_n \cdot \boldsymbol{p}_n$ as zero. A small but non-zero result might then be multiplied by $f'_n = \frac{(\widetilde{V}''_n l_n - \widetilde{V}'_n)}{l_n^2}$ in the computation of $\dot{f}_n = f'_n \dot{l}_n$, say: for the potential function given in (7.1), we have

$$\frac{(\widetilde{V}''_n l_n - \widetilde{V}'_n)}{l_n^2} = \frac{k l_n}{\bar{l}^2}$$

which is *large* for stiff problems. Therefore a small error in the computation of $\dot{l}_n$ will lead to a large value for $\dot{f}_n$ which should also be zero for steady-state problems. Hence the EMp schemes result in an *ill-conditioned* set of equations when applied to stiff steady-state problems.

We believe that this underlying fragility of the EMp schemes is one of the reasons for their poor performance with the stiff pendulum problem, which approximates the motion of a steady-state pendulum. We have also tested these algorithms on an actual stiff steady-state problem, and obtained results very similar to those given in Fig. 4. In particular, the explicit scheme EM10e did not preserve the relative equilibria for

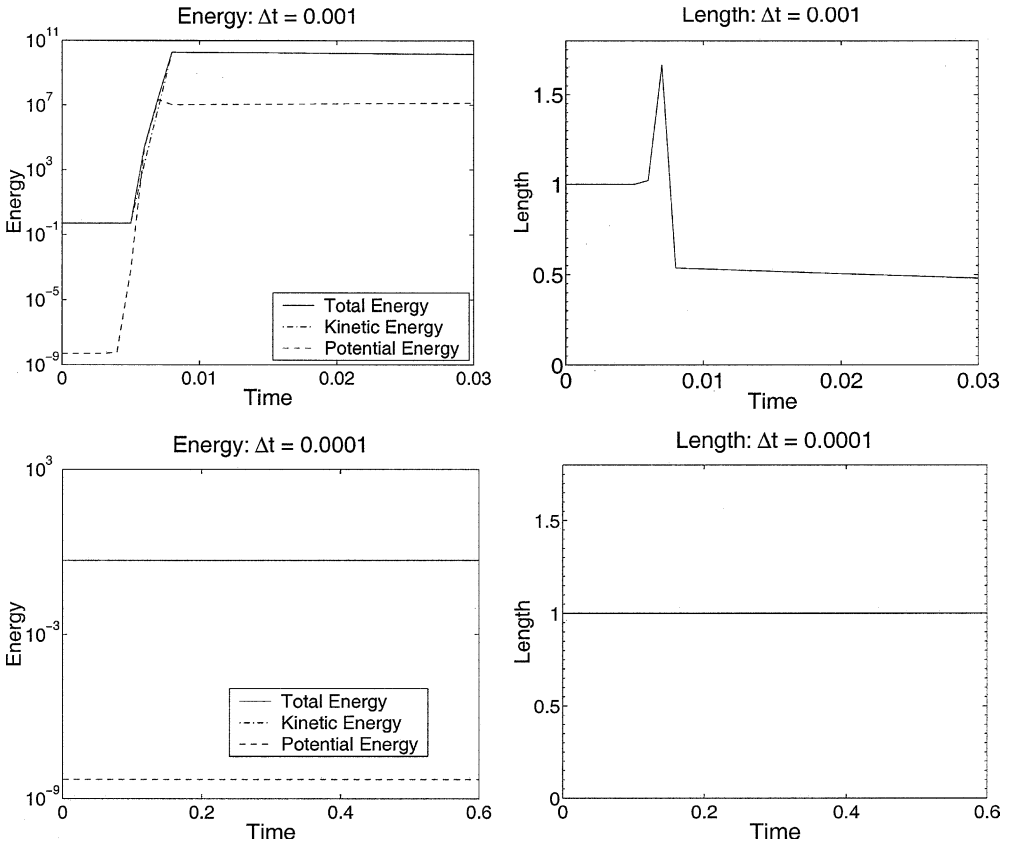

Fig. 7. Results obtained with EM10e for a stiff steady-state pendulum.

time-steps $\Delta t \geqslant 10^{-3}$ as shown in Fig. 7, which contradicts the theoretical guarantee given in Section 8.1. Similar anomalous behaviour has been observed for the well-known Trapezoidal Rule (or Average Acceleration Method; see e.g. [12]), which has also been proven to preserve relative equilibria [9]: earlier experiments, however, report the failure of the method when used for steady-state problems at relatively large time-steps [7,36]. In these experiments, the failure was characterised by a sharp increase in energy, leading to a loss of Newton–Raphson convergence: from Lemma 1, this would be impossible if the orbits of relative equilibria were perfectly preserved. These failures may be caused by round-off error, or the occurrence of non-unique solutions as mentioned in Proposition 2. For the explicit scheme EM10e, only one solution exists for $\boldsymbol{q}_{n+1}$, hence we attribute the failure of this algorithm to round-off error.

The marked improvement in Newton–Raphson convergence of EMTR4 as compared with EM4 can be attributed to both the absence of dot product calculations and also the time-reversibility of EMTR4. The smaller errors seen with EMTR4 may be due to the fact that β and γ now have infinite series expansions with respect to Δt, thus affecting error terms at $\mathcal{O}(\Delta t^5)$ and beyond, in contrast to EM4. However, the results of Section 8.2.2 indicate that improvements still need to be made in order to tackle stiff problems effectively. Examples such as the stiff pendulum have a large disparity between the periods of the high and low modes of motion, and generally need to be solved with a time-step size appropriate for the period of rotation, rather than that of vibration.

A further difficulty that can arise concerns the occurrence of solutions that are dependent on the choice of predictor used in the Newton–Raphson iteration, i.e. those that are not unique. The results in Section 8.2.1 suggest that the accuracy of the predictor may limit the accuracy of the resulting solution, thus higher-order predictors may be required for higher-order algorithms. Unfortunately, predictors more accurate than second order would involve expressions containing dot products, and thus be subject to round-off error.

8.4. Suggestions for improvement

For an algorithm to be truly efficient for a wide range of problems, it should combine higher-order accuracy with an ability to produce a reasonably accurate solution using larger time-steps for stiff problems. One idea is to create a hybrid scheme that combines EMTR4 at small time-steps with an algorithm designed to solve stiff problems at large time-steps. Such an algorithm was given in [24], and corresponds to (3.6) with

$$\alpha = \frac{1}{2}, \quad \beta = \frac{\frac{1}{2}\theta}{\tan\left(\frac{1}{2}\theta\right)} \quad \text{and} \quad \xi \text{ given by (4.3)}, \tag{8.3}$$

where θ is the incremental angle between $\boldsymbol{q}_n$ and $\boldsymbol{q}_{n+1}$. This algorithm is energy- and momentum-conserving and time-reversible, and also recovers exact solutions to steady-state problems at all time-steps for which it converges [24]. It is second-order accurate in general due to $\gamma = 0$, and the range of incremental angles allowed by the algorithm is $[0, \pi)$. Note that $\beta \to 0$ if and only if $\theta \to \pi$, which implies roughly two time-steps per period of rotation; thus we may expect more robustness with this algorithm at larger time-steps. We call this algorithm EM2β; the linearisation details are provided in [24].

Fig. 8 contrasts the performance of EMTR4, EM2β and EMM for the stiff pendulum problem at time $t_n = 0.6$ s. (For the non-stiff pendulum, EM2β gives very similar results to EMM.) We see immediately that EM2β is much more robust than EMTR4, presumably on account of the definition of β as mentioned earlier. It also gives much more accurate results at larger time-steps than EMM, despite the order of accuracy being the same for each. This is because β in $(8.3)_2$ has been specifically designed to eliminate the error in the period of rotation [24], hence the errors for EM2β should always be smaller than those for EMM. In fact, at larger time-steps, EM2β actually gives a more accurate solution to the stiff problem than to the non-stiff problem, since the former is dominated by rotational motion. From Table 4 we see that the number of iterations for EM2β is larger than for EMM for the largest time-step $\Delta t = 10^{-1}$, as will be the case when the incremental angle θ gets closer to π.

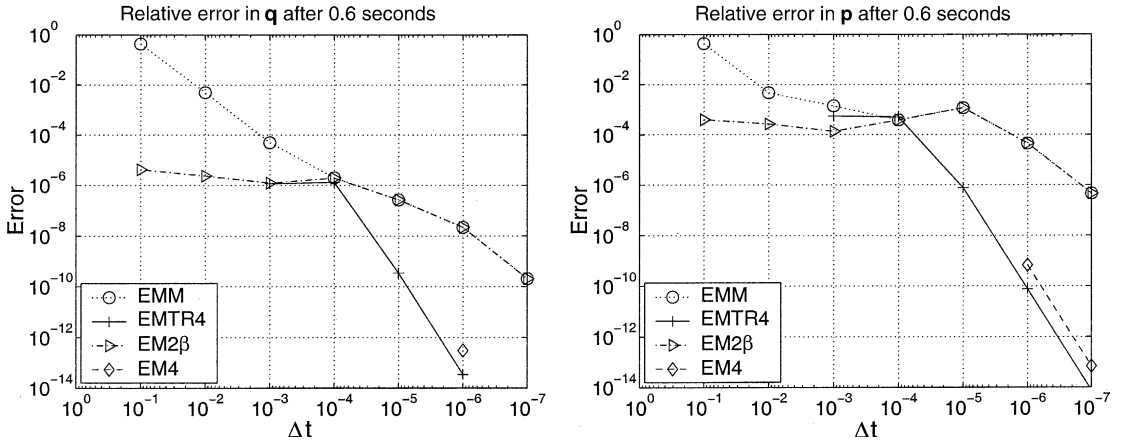

Fig. 8. Relative errors for the stiff pendulum.

Table 4
Average number of Newton–Raphson iterations needed per time-step for convergence for the stiff pendulum

Algorithm	Time-step size Δt						
	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
EMM	9	5	3	2	2	2	1
EM2β	27	5	3	2	2	2	1
EM4	*	*	*	*	*	2	1
EMTR4	†	†	3	2	2	2	1

* Convergence not achieved within 50 iterations.
† Solution aborted when $\mathscr{D} < 10^{-20}$.

Based on the above results, our hybrid scheme should therefore try to emulate algorithm EMTR4 for non-stiff problems or when small time-steps are used, and algorithm EM2β otherwise. One criterion that may be used is the size of the *linear sampling frequency* $\Omega \equiv \sqrt{\frac{k}{m}}\Delta t$ [12]. Since this takes into account both the stiffness and the time-step size, we can set a condition such as

$$\begin{cases} \Omega \leqslant \Omega_{\mathrm{cr}}: & \text{select algorithm EMTR4,} \\ \Omega > \Omega_{\mathrm{cr}}: & \text{select algorithm EM2}\beta \end{cases}$$

for some fixed value Ω_{cr}. Since Ω does not change during the solution of the problem, this condition need be tested just once at the beginning of the analysis, thus making implementation very simple.

Regarding the value of Ω_{cr} we note that, for problems with small amplitudes of axial vibration (such as the stiff pendulum example here), $\sqrt{k/m}$ provides a good estimate of the (non-constant) frequency of vibration ω_{v}. Therefore the number of time-steps per period of vibration T_{v} is approximately $\frac{2\pi}{\Omega}$. We recall from Fig. 8 that algorithm EMTR4 begins to show fourth-order error decay around $\Delta t = 10^{-4}$ i.e. for about six time-steps per period of axial vibration. A heuristic choice of Ω_{cr} could thus be

$$\Omega_{\mathrm{cr}} = 1.$$

In other words, *assuming the problem is dominated by rotational motion, we select algorithm EM2β unless the time-step is small enough to resolve the vibrational mode, in which case we select algorithm EMTR4.*

Given that EMTR4 converges for $\Delta t \leqslant 5 \times 10^{-3}$, we see that our choice for Ω_{cr} is sufficiently cautious to ensure a solution should always be found for stiff problems with negligible axial motion, on the evidence

provided by the example here. We acknowledge that this criterion may not cater for problems with large amplitudes of axial vibration: for such problems, however, one should also acknowledge that the St. Venant–Kirchhoff potential used in these examples is not appropriate, since it does not grow sufficiently in the presence of large strains. In any case, for the non-stiff pendulum with $k = 10^2$ and $m = 1$, EMTR4 would be selected for all $\Delta t \leqslant 10^{-1}$ using this criterion, which reaps the benefits of fourth-order accuracy at all of the time-steps tested here.

In summary, we can say that based on the experimental evidence from our examples, the order of accuracy of a given algorithm is only significant if the time-step is sufficiently small to resolve *both* of the modes of motion. For larger time-steps, the global error in the resulting solution from a higher-order algorithm may in fact be equal to, or even greater than, that produced by a lower-order algorithm. Thus for problems where it is preferable to use time-steps appropriate to the lower mode of motion, algorithms should not be selected according to their order of accuracy p. Instead, they should be selected by considering their conservation properties and other characteristics known to influence the global error.

9. Conclusions

A general framework for single-step, momentum-conserving time-integration schemes for a central-force problem has been presented. Within this framework, conditions for energy conservation, preservation of physical relative equilibria and time reversibility have been established. Families of algorithms that fulfil all of these conditions can be generated that have two free parameters. Criteria for a given order of accuracy of these algorithms (in a non-linear setting) have been systematically determined. By choosing the free parameters to be certain functions of the configuration and the time-step (so that they are not constant), arbitrarily accurate conservative algorithms that preserve the orbits of relative equilibria can be obtained. This is done without need of additional degrees of freedom or extra stages of calculation. It has been shown that if either of the free parameters is constant, the accuracy is limited to second order. It has also been proven that algorithms of a given order of accuracy conserve the various constants of motion to the same order or higher.

Two groups of higher-order conservative algorithms were tested, along with three established second-order conservative schemes, on a stiff and a non-stiff model problem. The first group were not time-reversible, and required extensive use of arithmetic operations which are prone to round-off error. They performed well with the non-stiff problem, but suffered severe Newton–Raphson convergence difficulties when applied to the stiff problem. The occurrence of non-unique solutions was evident for these schemes. The second group of algorithms did not involve sensitive arithmetic operations, and were also time-reversible. These algorithms maintained good performance for the non-stiff problem, and significantly improved upon the robustness of the first group for the stiff problem: they were also more accurate than the first group in all cases. However, they did not provide solutions at such large time-step sizes as the standard energy–momentum algorithm.

A hybrid scheme has been proposed that combines a fourth-order time-reversible algorithm with a second-order algorithm designed to be effective for stiff problems at large time-steps. It has been demonstrated that a higher-order algorithm will not necessarily have a smaller global error than a lower-order algorithm for all time-step sizes. The order of accuracy seems to be apparent only in situations where the time-step size is small enough to resolve both modes of motion. For solving stiff problems with moderate or large time-steps, then, it seems appropriate to choose algorithms based on their conservation properties (and other factors influencing the global error) rather than the order of accuracy.

In our future work, we will try to extend the present results to problems of multi-particle dynamics using truss finite elements.

Appendix A. Proofs of results

A.1. Proof of Proposition 1 (Energy conservation)

We start by expressing H_{n+1} and H_n in terms of $\boldsymbol{q}_{n+1}$ and $\boldsymbol{q}_n$ only, and use parameters a, b, c and d for convenience. From (3.1) we can write

$$\begin{aligned} b\boldsymbol{p}_n &= \boldsymbol{q}_{n+1} - a\boldsymbol{q}_n, \\ b\boldsymbol{p}_{n+1} &= d\boldsymbol{q}_{n+1} - \underbrace{(ad-bc)}_{=1}\boldsymbol{q}_n, \end{aligned}$$

and thus

$$\begin{aligned} b^2\boldsymbol{p}_n \cdot \boldsymbol{p}_n &= \|\boldsymbol{q}_{n+1}\|^2 - 2a\boldsymbol{q}_{n+1}\cdot\boldsymbol{q}_n + a^2\|\boldsymbol{q}_n\|^2, \\ b^2\boldsymbol{p}_{n+1} \cdot \boldsymbol{p}_{n+1} &= d^2\|\boldsymbol{q}_{n+1}\|^2 - 2d\boldsymbol{q}_{n+1}\cdot\boldsymbol{q}_n + \|\boldsymbol{q}_n\|^2. \end{aligned}$$

Therefore we can write

$$\begin{aligned} b^2(H_{n+1}-H_n) &= b^2(\widetilde{V}_{n+1}-\widetilde{V}_n) + \frac{b^2}{2m}(\boldsymbol{p}_{n+1}\cdot\boldsymbol{p}_{n+1} - \boldsymbol{p}_n\cdot\boldsymbol{p}_n) \\ &= b^2\widetilde{V}_\Delta + \frac{1}{2m}[(d^2-1)\|\boldsymbol{q}_{n+1}\|^2 + 2(a-d)\boldsymbol{q}_{n+1}\cdot\boldsymbol{q}_n + (1-a^2)\|\boldsymbol{q}_n\|^2]. \end{aligned} \tag{A.1}$$

Provided that $b \neq 0$, we have energy conservation if and only if the right-hand side of (A.1) is zero. Using the parameter relations in (3.8) and multiplying (A.1) through by $(m/\Delta t)^2$ (which keeps the units as Joules, since b has units s/kg) and also by $\mathscr{D}^2$ results in (4.2), after some manipulation. (Note that $b \neq 0$ implies $\beta \neq 0$.) □

A.2. Proof of Proposition 2 (Preservation of relative equilibria)

Using (3.1) to express $\boldsymbol{q}_{n+1}$ and $\boldsymbol{p}_{n+1}$, we can write

$$\begin{aligned} \|\boldsymbol{q}_{n+1}\|^2 &= a^2\|\boldsymbol{q}_n\|^2 + 2ab\boldsymbol{q}_n\cdot\boldsymbol{p}_n + b^2\|\boldsymbol{p}_n\|^2, \\ \|\boldsymbol{p}_{n+1}\|^2 &= c^2\|\boldsymbol{q}_n\|^2 + 2cd\boldsymbol{q}_n\cdot\boldsymbol{p}_n + d^2\|\boldsymbol{p}_n\|^2 \quad \text{and} \\ \boldsymbol{q}_{n+1}\cdot\boldsymbol{p}_{n+1} &= ac\|\boldsymbol{q}_n\|^2 + (ad+bc)\boldsymbol{q}_n\cdot\boldsymbol{p}_n + bd\|\boldsymbol{p}_n\|^2. \end{aligned}$$

If we now impose the relative equilibrium conditions at time-step n, namely $\boldsymbol{q}_n\cdot\boldsymbol{p}_n = 0$ and $\frac{1}{m}\|\boldsymbol{p}_n\|^2 = f_n\|\boldsymbol{q}_n\|^2$, this expression reduces to

$$\begin{aligned} \|\boldsymbol{q}_{n+1}\|^2 &= (a^2 + b^2mf_n)\|\boldsymbol{q}_n\|^2, \\ mf_n\|\boldsymbol{p}_{n+1}\|^2 &= (c^2 + d^2mf_n)\|\boldsymbol{p}_n\|^2 \quad \text{and} \\ \boldsymbol{q}_{n+1}\cdot\boldsymbol{p}_{n+1} &= (ac + bdmf_n)\|\boldsymbol{q}_n\|^2. \end{aligned} \tag{A.2}$$

For a relative equilibrium solution at time-step $n+1$ to be *possible*, we must therefore have

$$\begin{aligned} &(a^{\mathrm{RE}})^2 + (b^{\mathrm{RE}})^2mf_n = 1, \quad (c^{\mathrm{RE}})^2 + (d^{\mathrm{RE}})^2mf_n = mf_n \quad \text{and} \\ &a^{\mathrm{RE}}c^{\mathrm{RE}} + b^{\mathrm{RE}}d^{\mathrm{RE}}mf_n = 0 \end{aligned} \tag{A.3}$$

as seen by inserting the relative equilibrium solution $\|\boldsymbol{q}_{n+1}\| = \|\boldsymbol{q}_n\|$, $\|\boldsymbol{p}_{n+1}\| = \|\boldsymbol{p}_n\|$ and $\boldsymbol{q}_{n+1} \cdot \boldsymbol{p}_{n+1} = 0$ into (A.2). Multiplying $(A.3)_1$ by d^{RE} and $(A.3)_2$ by a^{RE} and incorporating $(A.3)_3$ into each leads to

$$d^{RE}(a^{RE})^2 - b^{RE}a^{RE}c^{RE} = d^{RE} \quad \text{and} \quad -c^{RE}b^{RE}d^{RE}mf_n + a^{RE}(d^{RE})^2 mf_n = a^{RE}mf_n$$

which, when using the fact that $ad - bc = 1$ for momentum-conserving schemes, gives us the single condition

$$a^{RE} = d^{RE}. \tag{A.4}$$

Putting this back into $(A.3)_3$ yields

$$c^{RE} + b^{RE}mf_n = 0. \tag{A.5}$$

Note that (A.5) holds even for $a^{RE} = d^{RE} = 0$, since then $a^{RE}d^{RE} - b^{RE}c^{RE} = 1$ reduces to $-b^{RE}c^{RE} = 1$. Using the parameter relations in (3.8) and multiplying through by $\mathscr{D}$ results in (4.8).

These two equations imply that a relative equilibrium solution is a possible configuration for $\boldsymbol{p}_{n+1}$ and $\boldsymbol{q}_{n+1}$; they do not imply that it is the *only* configuration. If, however, the algorithm gives a unique solution for $\boldsymbol{p}_{n+1}$ and $\boldsymbol{q}_{n+1}$, then we are assured that it will be the relative equilibrium solution. □

A.3. Proof of Proposition 3 (Time reversibility)

From (3.2), (4.14) and (4.15) we see that Algorithm 1 is time-reversible if and only if

$$\boldsymbol{B}_{n+1}^{TR} = \boldsymbol{B}_{n+1}^{-1}.$$

Using (3.5), this leads to

$$a^{TR} = d, \quad b^{TR} = -b, \quad c^{TR} = -c, \quad \text{and} \quad d^{TR} = a. \tag{A.6}$$

Inserting the parameter relations in (3.8) into (A.6) and solving the resulting equations for β, γ and ξ leads to (4.16), after some manipulation. □

A.4. Proof of Theorem 1

If an algorithm is pth-order accurate then from (5.1) we know that

$$\boldsymbol{\epsilon} = \boldsymbol{z}_{n+1} - \boldsymbol{z}(t_{n+1}) \in \mathcal{O}(\Delta t^{p+1}) \tag{A.7}$$

when $\boldsymbol{z}_n = \boldsymbol{z}(t_n)$. We now define $\boldsymbol{\tau}, \boldsymbol{\sigma} \in \mathbb{R}^3$ such that

$$\boldsymbol{\tau} = \boldsymbol{q}_{n+1} - \boldsymbol{q}(t_{n+1}) \quad \text{and} \quad \boldsymbol{\sigma} = \boldsymbol{p}_{n+1} - \boldsymbol{p}(t_{n+1}),$$

thus $\boldsymbol{\epsilon}^T = [\,\boldsymbol{\tau}^T \quad \boldsymbol{\sigma}^T\,]$ where $\boldsymbol{\tau}$ and $\boldsymbol{\sigma}$ are $\mathcal{O}(\Delta t^{p+1})$ for a pth-order algorithm.

For $H_{n+1} \equiv H(\boldsymbol{z}_{n+1})$ we can write

$$H_{n+1} = H[\boldsymbol{z}(t_{n+1}) + \boldsymbol{\epsilon}] = H[\boldsymbol{z}(t_{n+1})] + \nabla H[\boldsymbol{z}(t_{n+1})] \cdot \boldsymbol{\epsilon} + \mathcal{O}(\|\boldsymbol{\epsilon}\|^2)$$

using Taylor's theorem. Thus the local energy error is

$$\varepsilon^H = H_{n+1} - H(t_{n+1}) = \nabla_z H[\boldsymbol{z}(t_{n+1})] \cdot \boldsymbol{\epsilon} + \mathcal{O}(\|\boldsymbol{\epsilon}\|^2), \tag{A.8}$$

therefore ε^H can be at most $\mathcal{O}(\Delta t^{p+1})$ due to (A.7).

For the local angular momentum error, we have

$$\begin{aligned}\boldsymbol{\varepsilon}^{\mathscr{J}} &= \boldsymbol{q}_{n+1} \times \boldsymbol{p}_{n+1} - \boldsymbol{q}(t_{n+1}) \times \boldsymbol{p}(t_{n+1}) = (\boldsymbol{q}(t_{n+1}) + \boldsymbol{\tau}) \times (\boldsymbol{p}(t_{n+1}) + \boldsymbol{\sigma}) - \boldsymbol{q}(t_{n+1}) \times \boldsymbol{p}(t_{n+1}) \\ &= \boldsymbol{\tau} \times \boldsymbol{p}(t_{n+1}) + \boldsymbol{q}(t_{n+1}) \times \boldsymbol{\sigma} + \boldsymbol{\tau} \times \boldsymbol{\sigma}\end{aligned}$$

which is at most $\mathcal{O}(\Delta t^{p+1})$.

For the local error in the orbits of relative equilibria, we have

$$\|\boldsymbol{q}_{n+1}\|^2 = \boldsymbol{q}(t_{n+1}) \cdot \boldsymbol{q}(t_{n+1}) + 2\boldsymbol{q}(t_{n+1}) \cdot \boldsymbol{\tau} + \|\boldsymbol{\tau}\|^2,$$
$$\|\boldsymbol{p}_{n+1}\|^2 = \boldsymbol{p}(t_{n+1}) \cdot \boldsymbol{p}(t_{n+1}) + 2\boldsymbol{p}(t_{n+1}) \cdot \boldsymbol{\sigma} + \|\boldsymbol{\sigma}\|^2 \quad \text{and}$$
$$\boldsymbol{q}_{n+1} \cdot \boldsymbol{p}_{n+1} = \boldsymbol{q}(t_{n+1}) \cdot \boldsymbol{p}(t_{n+1}) + \boldsymbol{p}(t_{n+1}) \cdot \boldsymbol{\tau} + \boldsymbol{q}(t_{n+1}) \cdot \boldsymbol{\sigma} + \boldsymbol{\tau} \cdot \boldsymbol{\sigma}$$

which shows that $\boldsymbol{q}_{n+1} \cdot \boldsymbol{p}_{n+1} - \boldsymbol{q}(t_{n+1}) \cdot \boldsymbol{p}(t_{n+1})$ is at most $\mathcal{O}(\Delta t^{p+1})$, and also

$$\|\boldsymbol{q}_{n+1}\| = \sqrt{\|\boldsymbol{q}(t_{n+1})\|^2 + 2\boldsymbol{q}(t_{n+1}) \cdot \boldsymbol{\tau} + \|\boldsymbol{\tau}\|^2} = \|\boldsymbol{q}(t_{n+1})\| + \mathcal{O}(\|\boldsymbol{\tau}\|) \quad \text{and}$$
$$\|\boldsymbol{p}_{n+1}\| = \sqrt{\|\boldsymbol{p}(t_{n+1})\|^2 + 2\boldsymbol{p}(t_{n+1}) \cdot \boldsymbol{\sigma} + \|\boldsymbol{\sigma}\|^2} = \|\boldsymbol{p}(t_{n+1})\| + \mathcal{O}(\|\boldsymbol{\sigma}\|)$$

where we have used the binomial expansion of the square root terms. Thus $\boldsymbol{\varepsilon}^R$ is at most $\mathcal{O}(\Delta t^{p+1})$. □

A.5. Proof of Theorem 2

Suppose that β and γ fulfil the conditions in Table 1 for pth-order accuracy where $p \geqslant 1$. Take $\xi = \hat{\xi}$ such that $\varepsilon^H \in \mathcal{O}(\Delta t^{p+p_1})$ for $p_1 \geqslant 1$. Since $\varepsilon^H = H_{n+1} - H(t_{n+1})$ when $\boldsymbol{z}_n = \boldsymbol{z}(t_n)$, and also $H(t_{n+1}) = H(t_n)$ for the exact solution, we have $\varepsilon^H = H_{n+1} - H_n$ and hence

$$H_\Delta(\hat{\xi}) \in \mathcal{O}(\Delta t^{p+p_1}),$$

where $H_\Delta(\hat{\xi}) = H_{n+1} - H_n$ with $\xi = \hat{\xi}$ as above.

Now take $\xi = \bar{\xi}$ such that the conditions for pth-order accuracy are satisfied for β, γ and $\bar{\xi}$. Then we have $\varepsilon^H \in \mathcal{O}(\Delta t^{p+p_2})$ for $p_2 \geqslant 1$, from Theorem 1, and hence

$$H_\Delta(\bar{\xi}) \in \mathcal{O}(\Delta t^{p+p_2}).$$

From (3.8) and (A.1), we have

$$\beta^2 H_\Delta(\xi) = A\xi + B$$

where $A = -\beta(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}) \cdot \boldsymbol{q}_{1/2}$ and $B = \beta^2 \widetilde{V}_\Delta - \frac{m\beta\gamma}{\Delta t^2}\|\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}\|^2$. Since $\beta_0 = 1$ from Table 1, we have $\beta \in \mathcal{O}(1)$ and thus

$$A\hat{\xi} + B \in \mathcal{O}(\Delta t^{p+p_1}) \quad \text{and} \quad A\bar{\xi} + B \in \mathcal{O}(\Delta t^{p+p_2}).$$

Subtracting the second from the first leads to

$$A(\hat{\xi} - \bar{\xi}) \in \mathcal{O}(\Delta t^{\mathrm{Min}\{p+p_1, p+p_2\}}). \tag{A.9}$$

Now, since $\beta \in \mathcal{O}(1)$ and $\gamma \in \mathcal{O}(\Delta t^2)$ from the accuracy requirements, $A \in \mathcal{O}(\Delta t)$ in general; hence (A.9) gives

$$\hat{\xi} - \bar{\xi} \in \mathcal{O}(\Delta t^r)$$

where $r = \mathrm{Min}\{p + p_1, p + p_2\} - 1$. Thus $r \geqslant p$, and using the power series expansion for ξ in terms of Δt given in (5.16), we have

$$\hat{\xi}_s = \bar{\xi}_s, \quad s = 0, \ldots, p-1.$$

Since $\bar{\xi}$ was chosen to satisfy the requirements for pth-order accuracy in Table 1, this means that $\hat{\xi}$, in conjunction with β and γ, is sufficient for pth-order accuracy. □

Appendix B. Newton–Raphson linearisation

Recall from (5.2) the residual vector

$$\boldsymbol{g}(\boldsymbol{x}) := \widehat{\boldsymbol{B}}(\boldsymbol{x}, \boldsymbol{z}_n, \Delta t)\boldsymbol{z}_n - \mathscr{D}(\boldsymbol{x}, \boldsymbol{z}_n, \Delta t)\boldsymbol{x},$$

where $\mathscr{D}$ and $\widehat{\boldsymbol{B}}$ are defined in (3.8) and (3.9). Thus determining $\boldsymbol{q}_{n+1}$ and $\boldsymbol{p}_{n+1}$ from Algorithm 1 is tantamount to solving the non-linear equation

$$\boldsymbol{g}(\boldsymbol{z}_{n+1}) = \boldsymbol{0}, \tag{B.1}$$

where $\boldsymbol{z}^{\mathrm{T}} = [\,\boldsymbol{q}^{\mathrm{T}} \quad \boldsymbol{p}^{\mathrm{T}}\,]$. We solve this using a standard Newton–Raphson iteration (see e.g. [21]) where the Jacobian matrix

$$\boldsymbol{K}(\boldsymbol{z}_{n+1}) \equiv \nabla\boldsymbol{g} \equiv \boldsymbol{g} \otimes \nabla \tag{B.2}$$

has components $K_{ij} = \frac{\partial g_i}{\partial x_j}$. By writing $\boldsymbol{g}^{\mathrm{T}} = [\,\boldsymbol{g}_1^{\mathrm{T}} \quad \boldsymbol{g}_2^{\mathrm{T}}\,]$ where

$$\begin{aligned} \boldsymbol{g}_1(\boldsymbol{z}_{n+1}) &:= a\mathscr{D}\boldsymbol{q}_n + b\mathscr{D}\boldsymbol{p}_n - \mathscr{D}\boldsymbol{q}_{n+1}, \\ \boldsymbol{g}_2(\boldsymbol{z}_{n+1}) &:= c\mathscr{D}\boldsymbol{q}_n + d\mathscr{D}\boldsymbol{p}_n - \mathscr{D}\boldsymbol{p}_{n+1}, \end{aligned} \tag{B.3}$$

and noting that a, b, c, d and $\mathscr{D}$ are functions of $\boldsymbol{q}_n$, $\boldsymbol{p}_n$, $\boldsymbol{q}_{n+1}$ and Δt only, we see that (B.1) is *linear* in $\boldsymbol{p}_{n+1}$. Thus (B.2) becomes

$$\boldsymbol{K}(\boldsymbol{z}_{n+1}) \equiv \begin{pmatrix} \boldsymbol{K}_{11} & \boldsymbol{K}_{12} \\ \boldsymbol{K}_{21} & \boldsymbol{K}_{22} \end{pmatrix} = \begin{pmatrix} \boldsymbol{g}_1 \otimes \nabla_{\boldsymbol{q}_{n+1}} & \boldsymbol{0}_3 \\ \boldsymbol{g}_2 \otimes \nabla_{\boldsymbol{q}_{n+1}} & -\mathscr{D}\boldsymbol{I}_3 \end{pmatrix}$$

and the Newton–Raphson procedure amounts to solving

$$\boldsymbol{K}_{11}\Delta\boldsymbol{q}_{n+1}^{\langle i\rangle} = -\boldsymbol{g}_1$$

for the incremental position update $\Delta\boldsymbol{q}_{n+1}^{\langle i\rangle}$, which is then used to update $\boldsymbol{q}_{n+1}$ via $\boldsymbol{q}_{n+1}^{\langle i+1\rangle} := \boldsymbol{q}_{n+1}^{\langle i\rangle} + \Delta\boldsymbol{q}_{n+1}^{\langle i\rangle}$. This process is repeated until $\|\boldsymbol{g}_1(\boldsymbol{q}_{n+1}^{\langle i+1\rangle})\| < \epsilon\|\boldsymbol{q}_0\|$ for a prescribed tolerance ϵ, at which point we define $\boldsymbol{q}_{n+1} := \boldsymbol{q}_{n+1}^{\langle i+1\rangle}$ and insert this value into $(3.2)_1$ to recover $\boldsymbol{p}_{n+1}$. From (B.3) we have

$$\boldsymbol{K}_{11} \equiv \boldsymbol{g}_1 \otimes \nabla_{\boldsymbol{q}_{n+1}} = \boldsymbol{q}_n \otimes \nabla_{\boldsymbol{q}_{n+1}}\{a\mathscr{D}\} + \boldsymbol{p}_n \otimes \nabla_{\boldsymbol{q}_{n+1}}\{b\mathscr{D}\} - \mathscr{D}\boldsymbol{I}_3 - \boldsymbol{q}_{n+1} \otimes \nabla_{\boldsymbol{q}_{n+1}}\mathscr{D}. \tag{B.4}$$

We can express $\boldsymbol{K}_{11}$ in terms of the parameters β, γ and ξ by first recalling that $\mathscr{D} = \beta^2 - \frac{1}{4}\gamma^2 + \frac{1}{4m}\xi\Delta t^2$ and using (3.8) to get

$$\begin{aligned} &\nabla\{a\mathscr{D}\} = (2\beta+\gamma)\left(\nabla\beta + \frac{1}{2}\nabla\gamma\right) - \frac{1}{4m}\Delta t^2\nabla\xi, \quad \nabla\{b\mathscr{D}\} = \frac{1}{m}\Delta t\nabla\beta \quad \text{and} \\ &\nabla\mathscr{D} = 2\beta\nabla\beta - \frac{1}{2}\gamma\nabla\gamma + \frac{1}{4m}\Delta t^2\nabla\xi, \end{aligned} \tag{B.5}$$

where the symbol ∇ now denotes $\nabla_{\boldsymbol{q}_{n+1}}$. Now inserting the definitions from (B.5) into (B.4) gives us, after rearranging,

$$\begin{aligned} \boldsymbol{K}_{11} = &\left[(2\beta+\gamma)\boldsymbol{q}_n + \frac{1}{m}\Delta t\boldsymbol{p}_n - 2\beta\boldsymbol{q}_{n+1}\right] \otimes \nabla\beta + \left[\left(\beta + \frac{1}{2}\gamma\right)\boldsymbol{q}_n - \frac{1}{2}\gamma\boldsymbol{q}_{n+1}\right] \otimes \nabla\gamma \\ &- \frac{1}{4m}\Delta t^2(\boldsymbol{q}_n + \boldsymbol{q}_{n+1}) \otimes \nabla\xi - \mathscr{D}\boldsymbol{I}_3. \end{aligned} \tag{B.6}$$

B.1. EMTR4

From (8.1) we have

$$\nabla\beta = \left(\frac{\sin\left(\sqrt{\frac{f_{1/2}}{m}}\Delta t\right) - \sqrt{\frac{f_{1/2}}{m}}\Delta t}{\sqrt{\frac{f_{1/2}}{m}}\Delta t\left[1-\cos\left(\sqrt{\frac{f_{1/2}}{m}}\Delta t\right)\right]} \right) \frac{\Delta t^2}{8m}\frac{f'_{n+1}}{l_{n+1}}\boldsymbol{q}_{n+1}. \tag{B.7}$$

When $\sqrt{\frac{f_{1/2}}{m}}\Delta t = 0$ this becomes $\nabla\beta = -\frac{\Delta t^2}{24m}\frac{f'_{n+1}}{l_{n+1}}\boldsymbol{q}_{n+1}$, since $\lim_{\theta\to 0}\left\{\frac{\sin\theta-\theta}{\theta(1-\cos\theta)}\right\} = -\frac{1}{3}$. Similarly,

$$\nabla\gamma = \frac{\Delta t^2}{12m}\nabla f_{n+1} = \frac{\Delta t^2}{12m}\frac{f'_{n+1}}{l_{n+1}}\boldsymbol{q}_{n+1}. \tag{B.8}$$

Thus with ξ given by (4.2), we have

$$\nabla\xi = \frac{\widetilde{V}_\Delta\nabla\beta + \beta\nabla\widetilde{V}_\Delta - \frac{m}{\Delta t^2}\left[\|\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}\|^2\nabla\gamma + \gamma\nabla\{\|\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}\|^2\}\right]}{Y} - \frac{\xi}{Y}\nabla\{(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2})\cdot\boldsymbol{q}_{1/2}\}$$

with $Y = (\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2})\cdot\boldsymbol{q}_{1/2}$. We also have

$$\nabla\widetilde{V}_\Delta = f_{n+1}\boldsymbol{q}_{n+1},$$

$$\nabla\{\|\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}\|^2\} = (2\beta-\gamma)(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}) + 2[\boldsymbol{q}_\Delta\cdot(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2})]\nabla\beta - 2[\boldsymbol{q}_{1/2}\cdot(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2})]\nabla\gamma$$

and

$$\nabla\{(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2})\cdot\boldsymbol{q}_{1/2}\} = \beta\boldsymbol{q}_{n+1} - \gamma\boldsymbol{q}_{1/2} + (\boldsymbol{q}_\Delta\cdot\boldsymbol{q}_{1/2})\nabla\beta - (\boldsymbol{q}_{1/2}\cdot\boldsymbol{q}_{1/2})\nabla\gamma$$

which leads to

$$\begin{aligned}\nabla\xi = \frac{1}{Y}\Big[&\beta f_{n+1}\boldsymbol{q}_{n+1} - \frac{m}{\Delta t^2}\gamma(2\beta-\gamma)(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}) - \xi(\beta\boldsymbol{q}_{n+1} - \gamma\boldsymbol{q}_{1/2})\\ &+ (\widetilde{V}_\Delta + 2\gamma[\boldsymbol{q}_\Delta\cdot(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2})] - \xi(\boldsymbol{q}_\Delta\cdot\boldsymbol{q}_{1/2}))\nabla\beta\\ &- \Big(\frac{m}{\Delta t^2}(\beta\boldsymbol{q}_\Delta - 3\gamma\boldsymbol{q}_{1/2})\cdot(\beta\boldsymbol{q}_\Delta - \gamma\boldsymbol{q}_{1/2}) - \xi(\boldsymbol{q}_{1/2}\cdot\boldsymbol{q}_{1/2})\Big)\nabla\gamma\Big].\end{aligned} \tag{B.9}$$

$\boldsymbol{K}_{11}$ is then obtained from (B.6)–(B.9).

B.2. EMp schemes

Here β and γ are defined entirely by quantities at time-step n, thus $\nabla\beta = \nabla\gamma = \boldsymbol{0}$ and the corresponding terms in (B.9) vanish. Substituting this expression for $\nabla\xi$, along with $\nabla\beta = \nabla\gamma = \boldsymbol{0}$, into (B.6) then gives $\boldsymbol{K}_{11}$.

References

[1] R.A. LaBudde, D. Greenspan, Energy and momentum conserving methods of arbitrary order for the numerical integration of equations of motion. I. Motion of a single particle, Numer. Math. 25 (1976) 323–346.

[2] A.J. Chorin, T.J.R. Hughes, M. McCracken, J.E. Marsden, Product formulas and numerical algorithms, Comm. Pure Appl. Math. 31 (1978) 205–256.

[3] J.C. Simo, N. Tarnow, The discrete energy–momentum method. Conserving algorithms for non-linear elastodynamics, J. Appl. Math. Phys. (ZAMP) 43 (1992) 757–792.

[4] J.C. Simo, N. Tarnow, K.K. Wong, Exact energy–momentum conserving algorithms and symplectic schemes for non-linear dynamics, Comput. Methods Appl. Mech. Engrg. 100 (1992) 63–116.

[5] J.C. Simo, O. Gonzalez, Assessment of energy–momentum and symplectic schemes for stiff dynamical systems. Papers—American Society of Mechanical Engineers—All Series, 93(4), 1993. Presented at the ASME Winter Annual Meeting, New Orleans, LA, November 28–December 3, 1993.
[6] D. Greenspan, Completely conservative, covariant numerical methodology, Comput. Math. Appl. 29 (4) (1995) 37–43.
[7] M.A. Crisfield, J. Shi, An energy conserving co-rotational procedure for nonlinear dynamics with finite elements, Nonlinear Dynamics 9 (1996) 37–52.
[8] S. Reich, Enhancing energy conserving methods, BIT 36 (1996) 122–134.
[9] F. Armero, I. Romero, On the formulation of high-frequency dissipative time-stepping algorithms for nonlinear dynamics. Part I: low order methods for two model problems and nonlinear elastodynamics, Comput. Methods Appl. Mech. Engrg. 190 (2001) 2603–2649.
[10] O. Gonzalez, J.C. Simo, On the stability of symplectic and energy–momentum algorithms for non-linear Hamiltonian systems with symmetry, Comput. Methods Appl. Mech. Engrg. 134 (1996) 197–222.
[11] G. Zhong, J.E. Marsden, Lie–Poisson Hamilton–Jacobi theory and Lie–Poisson integrators, Phys. Lett. A 133 (3) (1988) 134–139.
[12] T.J.R. Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Prentice-Hall, Englewood Cliffs, NJ, 1987.
[13] N. Tarnow, J.C. Simo, How to render second order accurate time-stepping algorithms fourth order accurate while retaining the stability and conservation properties, Comput. Methods Appl. Mech. Engrg. 115 (1994) 233–252.
[14] H. Yoshida, Construction of higher order symplectic integrators, Phys. Lett. A 150 (5) (1990) 262–268.
[15] E. Forest, Sixth-order Lie group integrators, J. Comput. Phys. 99 (1992) 209–213.
[16] T.C. Fung, Unconditionally stable higher-order Newmark methods by sub-stepping procedure, Comput. Methods Appl. Mech. Engrg. 147 (1997) 61–84.
[17] T.C. Fung, S.K. Chow, Solving non-linear problems by complex time step methods, Comm. Numer. Methods Engrg. 18 (2002) 287–303.
[18] P. Betsch, P. Steinmann, Conservation properties of a time FE method. Part I: time-stepping schemes for N-body problems, Int. J. Numer. Methods Engrg. 49 (2000) 599–638.
[19] S.C. Fan, T.C. Fung, G. Sheng, A comprehensive unified set of single-step algorithms with controllable dissipation for dynamics. Part I. Formulation, Comput. Methods Appl. Mech. Engrg. 145 (1997) 87–98.
[20] O.A. Bauchau, T. Joo, Computational schemes for non-linear elasto-dynamics, Int. J. Numer. Methods Engrg. 45 (1999) 693–719.
[21] A. Iserles, A First Course in the Numerical Analysis of Differential Equations, Cambridge University Press, Cambridge, England, 1996.
[22] J.H. Argyris, P.C. Dunne, T. Angelopoulos, Non-linear oscillations using the finite element technique, Comput. Methods Appl. Mech. Engrg. 2 (1973) 203–250.
[23] J.H. Argyris, P.C. Dunne, T. Angelopoulos, Dynamic response by large step integration, Earthquake Engineering and Structural Dynamics 2 (1973) 185–203.
[24] G. Jelenić, A family of implicit one-step algorithms for integrating motion in central force fields, submitted for publication, 2002.
[25] L. Collatz, The Numerical Treatment of Differential Equations, third ed., Springer-Verlag, Berlin, Germany, 1960.
[26] P. Henrici, Discrete Variable Methods in Ordinary Differential Equations, John Wiley, Chichester, England, 1962.
[27] E. Graham, G. Jelenić, Higher-order accuracy in conservative single-step time-integration schemes for a central-force system, Aero Report 2003-01, Imperial College of Science, Technology and Medicine, 2003.
[28] M.B. Monagan, K.O. Geddes, K.M. Heal, G. Labahn, S.M. Vorkoetter, J. McCarron, Maple 7 Programming Guide, Waterloo Maple, Waterloo, Ont., Canada, 2000.
[29] D. Lewis, J.C. Simo, Conserving algorithms for the dynamics of Hamiltonian systems on Lie groups, J. Nonlinear Sci. 4 (1994) 253–299.
[30] F. Kang, Difference schemes for Hamiltonian formalism and symplectic geometry, J. Comput. Math. 4 (1986) 279–289.
[31] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II, second ed., Springer-Verlag, Berlin, Germany, 1996.
[32] J.M. Sanz-Serna, M.P. Calvo, Numerical Hamiltonian Problems, Chapman and Hall, London, 1994.
[33] U.M. Ascher, S. Reich, On some difficulties in integrating highly oscillatory Hamiltonian systems, Lecture Notes in Comput. Sci. Engrg. 4 (1998) 281–296.
[34] E. Graham, G. Jelenić, M.A. Crisfield, A note on the equivalence of two recent time-integration schemes for N-body problems, Comm. Numer. Methods Engrg. 18 (2002) 615–620.
[35] F. Armero, E. Petőcz, Formulation and analysis of conserving algorithms for frictionless dynamic contact/impact problems, Comput. Methods Appl. Mech. Engrg. 158 (3–4) (1998) 269–300.
[36] D. Kuhl, M.A. Crisfield, Energy-conserving and decaying algorithms in nonlinear structural dynamics, Int. J. Numer. Methods Engrg. 45 (1999) 569–599.
[37] S. Linnainmaa, Software for doubled-precision floating-point computations, ACM Trans. Math. Software 7 (3) (1981) 272–283.